Available online at www.sciencedirect.com

SCIENCE DIRECT®

ELSEVIER

Theoretical Computer Science 360 (2006) 247–288

Theoretical Computer Science

www.elsevier.com/locate/tcs

Multimodal logic programming

Linh Anh Nguyen

Institute of Informatics, University of Warsaw, ul. Banacha 2, 02-097 Warsaw, Poland

Received 8 October 2003; received in revised form 13 October 2005; accepted 26 March 2006

Communicated by H. Katsuno

Abstract

We give a framework for developing the least model semantics, fixpoint semantics, and SLD-resolution calculi for logic programs in multimodal logics whose frame restrictions consist of the conditions of seriality (i.e. $\forall x\ \exists y\ R_i(x, y)$) and some classical first-order Horn clauses. Our approach is direct and no special restriction on occurrences of $\Box_i$ and $\Diamond_i$ is required. We apply our framework for a large class of basic serial multimodal logics, which are parameterized by an arbitrary combination of generalized versions of axioms T, B, 4, 5 (in the form, e.g. $4 : \Box_i\varphi \to \Box_j\Box_k\varphi$) and $I : \Box_i\varphi \to \Box_j\varphi$. Another part of the work is devoted to programming in multimodal logics intended for reasoning about multidegree belief, for use in distributed systems of belief, or for reasoning about epistemic states of agents in multiagent systems. For that we also use the framework, and although these latter logics belong to the mentioned class of basic serial multimodal logics, the special SLD-resolution calculi proposed for them are more efficient.

Keywords: Modal logic; Logic programming; Logics of belief; Kripke models; MProlog

1. Introduction

Classical logic programming is very useful in practice and has been thoroughly studied by many researchers. There are three standard semantics for definite logic programs: the least model semantics, the fixpoint semantics, and the SLD-resolution calculus (a procedural semantics) [26]. SLD-resolution, named by Apt and van Emden in [4], was first described by Kowalski [24] for logic programming. It is a top-down procedure for answering queries in definite logic programs. On the other hand, the fixpoint semantics of logic programs is a bottom-up method for answering queries and was first introduced by van Emden and Kowalski [41] using the direct consequence operator T_P. This operator is monotonic, continuous, and has the least fixpoint $T_P \uparrow \omega = \bigcup_{n=0}^{\omega} T_P \uparrow n$, which forms the least Herbrand model of the given logic program P.

Modal and temporal logics are useful in many areas of computer science. For example, multimodal logics are used in knowledge representation and multiagent systems by interpreting $\Box_i\varphi$ as "agent i knows/believes that φ is true". Many authors have proposed modal and temporal extensions for logic programming (see [40,20] for surveys [1]). There are two approaches to modal logic programming: the direct approach [18,6,10,31,32] and the translation approach [1,15,38]. The first approach directly uses modalities, while the second one translates modal logic programs to classical logic programs.

E-mail address: nguyen@mimuw.edu.pl.

[1] The works [38,10,31] on modal logic programming are not covered by the surveys.

0304-3975/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2006.03.026

In [15], Debart et al. applied a functional translation technique for logic programs in multimodal logics which have a finite number of modal operators $\Box_i$ and $\Diamond_i$ of any type among *KD*, *KT*, *KD*4, *KT*4, *KF* and interaction axioms of the form $\Box_i \varphi \to \Box_j \varphi$. The technique is similar to the one used in Ohlbach's resolution calculus for modal logics [39]. Extra parameters are added to predicate symbols to represent paths in the Kripke model, and special unification algorithms are used to deal with them.

In [38], Nonnengart proposed a semi-functional translation (which translates existential modal operators using functional translation and translates universal modal operators using relational translation). His approach uses accessibility relations for translated programs, but with optimized clauses for representing properties of the accessibility relations, and does not modify unification. Nonnengart [38] applied the approach for modal logic programs in all of the basic serial monomodal logics *KD*, *T*, *KDB*, *B*, *KD*4, *S*4, *KD*5, *KD*45, and *S*5. He also gave an example in a multimodal logic of type *KD*45.

The translation approach is attractive: just translate and it is done. However, the problem is more complicated. Modal logics add more nondeterminism to the search process, which cannot be eliminated but must be dealt with in some way. In the functional translation [15], the modified unification algorithm may return different most general unifiers (mgu's), which cause branching. In the semi-functional translation [38], additional nondeterminism is caused by clauses representing frame restrictions of the used modal logic. In the direct approach considered shortly, additional nondeterminism is caused by modal rules which are used as meta-clauses. In our opinion, the direct approach is worth to study, as it is one of the main approaches to deal with modalities and may result in a deeper analysis of the problem.

Using the direct approach for modal logic programming, Balbiani et al. [6] gave a declarative semantics and an SLD-resolution calculus for a class of logic programs in the monomodal logics *KD*, *T*, and *S*4. The work assumes that the modal operator $\Box$ does not occur in bodies of program clauses and goals. In [10], Baldoni et al. gave a framework for developing declarative and operational semantics for logic programs in multimodal logics which have axioms of the form $[t_1]\ldots[t_n]\varphi \to [s_1]\ldots[s_m]\varphi$, where $[t_i]$ and $[s_j]$ are universal modal operators indexed by terms t_i and s_j, respectively. In that work, existential modal operators are disallowed in programs and goals.

In [31], we developed a fixpoint semantics, the least model semantics, and an SLD-resolution calculus in a direct way for modal logic programs in all of the mentioned basic serial monomodal logics. We also extended the SLD-resolution calculus for the almost serial monomodal logics *KB*, *K*5, *K*45, and *KB*5. There are two important properties of our approach in [31]: no special restriction on occurrences of $\Box$ and $\Diamond$ is assumed (programs and goals are of a normal form but the language is as expressive as the general modal Horn fragment) and the semantics are formulated closely to the style of classical logic programming (as in Lloyd's book [26]).

One of the main goals of this work is to extend the results and generalize the methods of our mentioned work for multimodal logics. In this work, we give a framework for developing the least model semantics, fixpoint semantics, and SLD-resolution calculi for logic programs in multimodal logics whose frame restrictions consist of the conditions of seriality (i.e. $\forall x\ \exists y\ R_i(x, y)$) and some classical first-order Horn clauses. Our approach is direct and no special restriction on occurrences of $\Box_i$ and $\Diamond_i$ is assumed. We prove that under certain expected properties of a concrete instantiation of the framework for a specific multimodal logic, the SLD-resolution calculus is sound and complete.

We apply our framework for a large class of basic serial multimodal logics, which are parameterized by an arbitrary combination of generalized versions of axioms *T*, *B*, 4, 5 (in the form, e.g. $4 : \Box_i \varphi \to \Box_j \Box_k \varphi$) and $I : \Box_i \varphi \to \Box_j \varphi$. We prove that the instantiation for that class of logics is correct, i.e. the fixpoint semantics coincides with the least model semantics, and the SLD-resolution calculus is sound and complete.

Another part of this work is devoted to programming in multimodal logics intended for reasoning about multidegree belief, for use in distributed systems of belief, or for reasoning about epistemic states of agents in multiagent systems. For that we also use the framework, and although these latter logics belong to the mentioned class of basic serial multimodal logics, the special SLD-resolution calculi proposed for them are more efficient.

To illustrate our approach of defining semantics for multimodal logic programs, we give here an example. Let the base logic be the simplest serial multimodal logic $KD_{(m)}$ and *P* be the following program:

$$\begin{aligned}
\varphi_1 &= \Diamond_1 p(a) \leftarrow \\
\varphi_2 &= \Box_1(\Box_2 q(x) \leftarrow p(x)) \\
\varphi_3 &= \Box_1(\Diamond_2 r(x) \leftarrow p(x), \Box_2 q(x)) \\
\varphi_4 &= \Box_1 \Box_2(s(x) \leftarrow q(x), r(x)) \\
\varphi_5 &= \Box_1(t(x) \leftarrow \Diamond_2 s(x)).
\end{aligned}$$

Goals	Input clauses	mgu's
$G = \leftarrow \Diamond_1 t(x)$		
$G' = \leftarrow \langle X\rangle_1 t(x)$		
$G_1 = \leftarrow \langle X\rangle_1 \Diamond_2 s(x)$	$\Box_1(t(x_1) \leftarrow \Diamond_2 s(x_1))$	$\{x_1/x\}$
$G_1' = \leftarrow \langle X\rangle_1 \langle Y\rangle_2 s(x)$		
$G_2 = \leftarrow \langle X\rangle_1\langle Y\rangle_2 q(x), \langle X\rangle_1\langle Y\rangle_2 r(x)$	$\Box_1\Box_2(s(x_2) \leftarrow q(x_2), r(x_2))$	$\{x_2/x\}$
$G_3 = \leftarrow \langle X\rangle_1\Box_2 q(x), \langle X\rangle_1\langle Y\rangle_2 r(x)$		
$G_4 = \leftarrow \langle X\rangle_1 p(x), \langle X\rangle_1\langle Y\rangle_2 r(x)$	$\Box_1(\Box_2 q(x_4) \leftarrow p(x_4))$	$\{x_4/x\}$
$G_5 = \leftarrow \langle p(a)\rangle_1\langle Y\rangle_2 r(a)$	$\Diamond_1 p(a) \leftarrow$	$\{x/a, X/p(a)\}$
$G_6 = \leftarrow \langle p(a)\rangle_1 p(a), \langle p(a)\rangle_1\Box_2 q(a)$	$\Box_1(\Diamond_2 r(x_6) \leftarrow p(x_6), \Box_2 q(x_6))$	$\{x_6/a, Y/r(a)\}$
$G_7 = \leftarrow \langle p(a)\rangle_1\Box_2 q(a)$	$\Diamond_1 p(a) \leftarrow$	ε
$G_8 = \leftarrow \langle p(a)\rangle_1 p(a)$	$\Box_1(\Box_2 q(x_8) \leftarrow p(x_8))$	$\{x_8/a\}$
the empty clause	$\Diamond_1 p(a) \leftarrow$	ε

Fig. 1. An illustrating example for SLD-resolution.

When building a $KD_{(m)}$-model graph M for P, to realize φ_1 at the actual world τ we connect τ to a world w via the accessibility relation R_1 and add $p(a)$ to w. The edge connecting τ to w is created due to $\Diamond_1 p(a)$, so we can label it by $\langle p(a)\rangle_1$ (a labeled form of $\Diamond_1$). The world w can be identified by τ and the edge from τ and denoted by the sequence $\tau\langle p(a)\rangle_1$. If we denote τ by the empty sequence then $w = \langle p(a)\rangle_1$. Apart from building M, we want to represent the model corresponding to M by a set I of *atoms*. To keep the information that $p(a)$ is true at w, we add the atom $\langle p(a)\rangle_1 p(a)$ to I. To realize φ_2 at τ, $\Box_2 q(x) \leftarrow p(x)$ is added to w, and then $\Box_2 q(a)$ is also added to w. To keep the fact that $\Box_2 q(a)$ belongs to w, we add $\langle p(a)\rangle_1\Box_2 q(a)$ to I. Note that I contains both $\langle p(a)\rangle_1 p(a)$ and $\langle p(a)\rangle_1\Box_2 q(a)$. Apply the rule φ_3 to I, then I should contain also $\langle p(a)\rangle_1\Diamond_2 r(a)$, which is then replaced by $\langle p(a)\rangle_1\langle r(a)\rangle_2 r(a)$ due to a similar reason as for φ_1. Since I contains both $\langle p(a)\rangle_1\Box_2 q(a)$ and $\langle p(a)\rangle_1\langle r(a)\rangle_2 r(a)$, after applying φ_4, I should contain also $\langle p(a)\rangle_1\langle r(a)\rangle_2 s(a)$. Finally, applying φ_5 to I, we get also $\langle p(a)\rangle_1 t(a)$. In general, instead of building a model graph for P we can build such a set I of atoms, which is called a *model generator*. The set $I_{KD_{(m)},P} = \{\langle p(a)\rangle_1 p(a), \langle p(a)\rangle_1\Box_2 q(a), \langle p(a)\rangle_1\langle r(a)\rangle_2 r(a), \langle p(a)\rangle_1\langle r(a)\rangle_2 s(a), \langle p(a)\rangle_1 t(a)\}$ is the least set of ground atoms which can be derived from P in $KD_{(m)}$ in this way. This set is obtained as the least fixpoint of a certain operator $T_{KD_{(m)},P}$ and is called the *least $KD_{(m)}$-model generator* of P.

Given a model generator I, we can construct the *standard $KD_{(m)}$-model* for it by building a model graph. During the construction, to realize a formula $\langle E\rangle_i\varphi$ at a world w, where E is a ground classical atom, we connect w via the accessibility relation R_i to the world identified by the sequence $w\langle E\rangle_i$ and add φ to that world. We realize a formula $\Box_i\varphi$ at a world w by adding φ to every world reachable from w via R_i. To guarantee the constructed model graph to be the smallest, each new world is connected via each accessibility relation to an empty world at the time of its creation. It can be shown that the standard $KD_{(m)}$-model of $I_{KD_{(m)},P}$ is a least $KD_{(m)}$-model of P.

Now let us give an SLD-refutation of $P \cup \{G\}$ in $KD_{(m)}$ for $G = \leftarrow \Diamond_1 t(x)$. By the content of $I_{KD_{(m)},P}$, the computed answer should be $\{x/a\}$. The SLD-refutation should trace back the process of deriving the atom $\langle p(a)\rangle_1 t(a)$ of $I_{KD_{(m)},P}$ from P. As a $KD_{(m)}$-resolvent of G and φ_5, we derive a new goal $G_1 = \leftarrow \Diamond_1\Diamond_2 s(x)$. As a $KD_{(m)}$-resolvent of G_1 and φ_4, we derive the goal $G_2 = \leftarrow \Diamond_1\Diamond_2(q(x) \wedge r(x))$. This goal is not desired, as it contains a formula but not atoms in its body. To overcome this problem, the (existential) modality $\Diamond_1\Diamond_2$ should be fixed first, e.g. to become $\langle X\rangle_1\langle Y\rangle_2$, then the goal G_2 can be rewritten to $\leftarrow \langle X\rangle_1\langle Y\rangle_2 q(x), \langle X\rangle_1\langle Y\rangle_2 r(x)$. The labeling should be done in two steps as follows: the goal $G = \leftarrow \Diamond_1 t(x)$ is first replaced by $G' = \leftarrow \langle X\rangle_1 t(x)$, the next goal in the derivation is $G_1 = \leftarrow \langle X\rangle_1\Diamond_2 s(x)$, which is then replaced by $G_1' = \leftarrow \langle X\rangle_1\langle Y\rangle_2 s(x)$, and then $G_2 = \leftarrow \langle X\rangle_1\langle Y\rangle_2 q(x), \langle X\rangle_1\langle Y\rangle_2 r(x)$ is derived from G_1' and φ_4. We can then strengthen G_2 to $G_3 = \leftarrow \langle X\rangle_1\Box_2 q(x), \langle X\rangle_1\langle Y\rangle_2 r(x)$. Resolving G_3 with φ_2, we obtain $G_4 = \leftarrow \langle X\rangle_1 p(x), \langle X\rangle_1\langle Y\rangle_2 r(x)$. Now resolve G_4 with φ_1. As explained in the construction of $I_{KD_{(m)},P}$, the atom $\Diamond_1 p(a)$ in the head of φ_1 can be treated as $\langle p(a)\rangle_1 p(a)$. Thus, resolving G_4 with φ_1 results in $G_5 = \leftarrow \langle p(a)\rangle_1\langle Y\rangle_2 r(a)$ and an mgu $\{x/a, X/p(a)\}$. Further steps are given in Fig. 1.

The rest of this work is organized as follows. In Section 2, we give basic definitions for multimodal logics, specify a class of basic serial multimodal logics, and introduce multimodal logics of belief. We also present an ordering of Kripke models and definitions involving with substitution and unification. In Section 3, we define the MProlog language for multimodal logic programming, which is as expressive as the general modal Horn fragment. Section 4 contains some examples of application of modal logic programming. In Section 5, we present our framework for developing semantics of MProlog programs in multimodal logics. The section starts with an introduction of labeled modal operators, their semantics, and notations that are used throughout the work. It then contains our formulations of the three mentioned semantics for MProlog programs. The section ends with a subsection concerning soundness and completeness of SLD-resolution. In Section 6, we instantiate the framework for the mentioned class of basic serial multimodal logics and prove its correctness. We continue such a task for multimodal logics of belief in Sections 7 and 8. In the last section, we discuss the relation to other works, describe the implemented modal logic programming system MProlog, and give some concluding remarks.

2. Preliminaries

2.1. Definitions for quantified multimodal logics

A language for quantified multimodal logics is an extension of the language of classical predicate logic with *modal operators* $\Box_i$ and $\Diamond_i$, for $1 \leqslant i \leqslant m$ (where m is a fixed number). If $m = 1$ then we ignore the subscript i and write $\Box$ and $\Diamond$. The modal operators $\Box_i$ and $\Diamond_i$ can take various meanings. For example, $\Box_i$ can stand for "the agent i believes" and $\Diamond_i$ for "it is considered possible by agent i". The operators $\Box_i$ are called *universal* modal operators, while $\Diamond_i$ are called *existential* modal operators.

Definition 2.1. A *term* is defined inductively as follows: a variable is a term; a constant symbol is a term; if f is an n-ary function symbol and $t_1, \ldots, t_n$ are terms, then $f(t_1, \ldots, t_n)$ is a term.

Definition 2.2. A (*well-formed modal*) *formula* is defined inductively as follows:
- If p is an n-ary predicate symbol and $t_1, \ldots, t_n$ are terms, then $p(t_1, \ldots, t_n)$ is a formula, called a *classical atom*.
- If φ and ψ are formulas, then so are $(\neg\varphi)$, $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, $(\varphi \to \psi)$, $(\Box_i \varphi)$, and $(\Diamond_i \psi)$.
- If φ is a formula and x is a variable, then $(\forall x.\varphi)$ and $(\exists x.\varphi)$ are formulas.

We also write $\varphi \equiv \psi$ for $(\varphi \to \psi) \wedge (\psi \to \varphi)$.

A term or a formula is *ground* if it does not contain variables.

If φ is a formula, then by $\forall(\varphi)$ we denote the *universal closure* of φ, which is the formula obtained by adding a universal quantifier for every variable having a free occurrence[2] in φ. Similarly, $\exists(\varphi)$ denotes the *existential closure* of φ, which is obtained by adding an existential quantifier for every variable having a free occurrence in φ.

The *modal depth* of a formula φ, denoted by $mdepth(\varphi)$, is the maximal nesting depth of modal operators occurring in φ. For example, the modal depth of $\Box_i(\Diamond_j p(x) \vee \Box_k q(y))$ is 2.

We now define Kripke models, model graphs, and the satisfaction relation.

Definition 2.3. A *Kripke frame* is a tuple $\langle W, \tau, R_1, \ldots, R_m \rangle$, where W is a nonempty set of possible worlds, $\tau \in W$ is the *actual world*, and R_i is a binary relation on W, called the *accessibility relation* for the modal operators $\Box_i$, $\Diamond_i$. If $R_i(w, u)$ holds then we say that the world u is accessible from the world w via R_i.

A frame $\langle W, \tau, R_1, \ldots, R_m \rangle$ is said to be *connected* if each of its worlds is directly or indirectly accessible from the actual world via the accessibility relations, i.e. for every $w \in W$ there exist $w_0 = \tau, w_1, \ldots, w_{k-1}, w_k = w$ with $k \geqslant 0$ such that $(w_i, w_{i+1}) \in R_1 \cup \cdots \cup R_m$ for all $0 \leqslant i < k$.

Definition 2.4. A *fixed-domain Kripke model with rigid terms*, hereafter simply called a Kripke model or just a model, is a tuple $M = \langle D, W, \tau, R_1, \ldots, R_m, \pi \rangle$, where D is a set called the *domain*, $\langle W, \tau, R_1, \ldots, R_m \rangle$ is a Kripke frame, and π is an interpretation of constant symbols, function symbols and predicate symbols. For a constant symbol a, $\pi(a)$ is an element of D, denoted by a^M. For an n-ary function symbol f, $\pi(f)$ is a function from D^n to D,

[2] I.e. an occurrence not bound by quantifiers.

denoted by f^M. For an n-ary predicate symbol p and a world $w \in W$, $\pi(w)(p)$ is an n-ary relation on D, denoted by $p^{M,w}$.

Definition 2.5. A *model graph* is a tuple $\langle W, \tau, R_1, \ldots, R_m, H\rangle$, where $\langle W, \tau, R_1, \ldots, R_m\rangle$ is a Kripke frame and H is a function that maps each world of W to a set of formulas.

Every model graph $\langle W, \tau, R_1, \ldots, R_m, H\rangle$ corresponds to an Herbrand model $M = \langle \mathcal{U}, W, \tau, R_1, \ldots, R_m, \pi\rangle$ specified by: $\mathcal{U}$ is the Herbrand universe (i.e. the set of all ground terms), $c^M = c$, $f^M(t_1, \ldots, t_n) = f(t_1, \ldots, t_n)$, and $((t_1, \ldots, t_n) \in p^{M,w}) \equiv (p(t_1, \ldots, t_n) \in H(w))$, where $t_1, \ldots, t_n$ are ground terms. We will sometimes treat a model graph as its corresponding model.

Definition 2.6. Let M be a Kripke model. A *variable assignment* (w.r.t. M) is a function that maps each variable to an element of the domain of M. The value of a term t w.r.t. a variable assignment V is denoted by $t^M[V]$ and defined as follows: if t is a constant symbol a then $t^M[V] = a^M$; if t is a variable x then $t^M[V] = V(x)$; if t is $f(t_1, \ldots, t_n)$ then $t^M[V] = f^M(t_1^M[V], \ldots, t_n^M[V])$.

Definition 2.7. Given some Kripke model $M = \langle D, W, \tau, R_1, \ldots, R_m, \pi\rangle$, some variable assignment V, and some world $w \in W$, the *satisfaction relation* $M, V, w \vDash \zeta$ for a formula ζ is defined as follows:

$$
\begin{array}{lll}
M, V, w \vDash p(t_1, \ldots, t_n) & \text{iff} & (t_1^M[V], \ldots, t_n^M[V]) \in p^{M,w}; \\
M, V, w \vDash \neg\varphi & \text{iff} & M, V, w \nvDash \varphi; \\
M, V, w \vDash \varphi \wedge \psi & \text{iff} & M, V, w \vDash \varphi \text{ and } M, V, w \vDash \psi; \\
M, V, w \vDash \varphi \vee \psi & \text{iff} & M, V, w \vDash \varphi \text{ or } M, V, w \vDash \psi; \\
M, V, w \vDash \varphi \to \psi & \text{iff} & M, V, w \nvDash \varphi \text{ or } M, V, w \vDash \psi; \\
M, V, w \vDash \Box_i \varphi & \text{iff} & \text{for all } v \in W \text{ such that } R_i(w, v),\ M, V, v \vDash \varphi; \\
M, V, w \vDash \Diamond_i \varphi & \text{iff} & \text{for some } v \in W,\ R_i(w, v) \text{ and } M, V, v \vDash \varphi; \\
M, V, w \vDash \forall x.\varphi & \text{iff} & \text{for all } a \in D,\ (M, V', w \vDash \varphi), \\
 & & \text{where } V'(x) = a \text{ and } V'(y) = V(y) \text{ for } y \neq x; \\
M, V, w \vDash \exists x.\varphi & \text{iff} & \text{there exists } a \in D \text{ such that } M, V', w \vDash \varphi, \\
 & & \text{where } V'(x) = a \text{ and } V'(y) = V(y) \text{ for } y \neq x.
\end{array}
$$

If $M, V, w \vDash \varphi$ then we say that φ is true at w in M w.r.t. V. We write $M, w \vDash \varphi$ to denote that $M, V, w \vDash \varphi$ for every V. We say that M satisfies φ, or φ is true in M, and write $M \vDash \varphi$, if $M, \tau \vDash \varphi$. For a set Γ of formulas, we call M a model of Γ and write $M \vDash \Gamma$ if $M \vDash \varphi$ for every $\varphi \in \Gamma$.

Let us explain why we include the actual world in the definition of Kripke models. Consider possible definitions of $M \vDash \Gamma$. Without the actual world one would define that $M \vDash \Gamma$ if $M, w \vDash \Gamma$ for every world w of M. This is not appropriate for our settings of modal logic programming: for example, when Γ is a logic program containing a classical fact $p(a)$, then we do not require that $p(a)$ is true at every possible world of M, because otherwise it would imply that $p(a)$ is "known" to be true in M.

A *logic* can be defined by a set of well-formed formulas, a class of admissible interpretations, and a satisfaction relation. The class of admissible interpretations for a modal logic L is often specified by restrictions on Kripke frames. We refer to such restrictions by *L-frame restrictions* and call frames with such properties *L-frames*.

Definition 2.8. We call a model M with an L-frame an *L-model*. We say that φ is *L-satisfiable* if there exists an L-model of φ, i.e. an L-model satisfying φ. A formula φ is said to be *L-valid* and called an *L-tautology* if φ is true in every L-model. For a set Γ of formulas, we write $\Gamma \vDash_L \varphi$ and call φ a *logical consequence* of Γ in L if φ is true in every L-model of Γ.

Note that our definition of $\Gamma \vDash_L \varphi$ reflects "local semantic consequence" due to the inclusion of actual world. Also note that $\Gamma \vDash_L \varphi$ means $\forall(\Gamma) \to \forall(\varphi)$ is an L-tautology.

If as the class of admissible interpretations we take the class of all Kripke models (with no restrictions on the accessibility relations) then we obtain the quantified multimodal logic $K_{(m)}$. This logic is axiomatized by the following system:

- axioms for classical predicate logic (without identity),
- the *K*-axioms: $\Box_i(\varphi \to \psi) \to (\Box_i\varphi \to \Box_i\psi)$,
- the Barcan formula axioms: $\forall x.\Box_i\varphi \to \Box_i\forall x.\varphi$,
- the axioms defining $\Diamond_i$: $\Diamond_i\varphi \equiv \neg\Box_i\neg\varphi$,
- the modus ponens rule: $\frac{\varphi \quad \varphi\to\psi}{\psi}$,
- the generalization rule: $\frac{\varphi}{\forall x.\varphi}$,
- and the modal generalization rules: $\frac{\varphi}{\Box_i\varphi}$.

Note that the converse Barcan formula $\Box_i\forall x.\varphi \to \forall x.\Box_i\varphi$ is a consequence of this axiomatization system. Every logic whose axiomatization is an extension of the system $K_{(m)}$ is called a *normal multimodal logic*.

2.2. A class of basic serial multimodal logics

A normal multimodal logic can be characterized by axioms extending the system $K_{(m)}$. Consider the class *BSMM* of basic serial multimodal logics specified as follows. A *BSMM* logic is a normal multimodal logic parameterized by relations $AD/1$, $AT/1$, $AI/2$, $AB/2$, $A4/3$, $A5/3$ on the set $\{1, \dots, m\}$, where the numbers on the right are arities and *AD* is required to be full (i.e. $AD(i)$ holds for every $1 \leqslant i \leqslant m$). These relations specify the following axioms:

$$\begin{array}{ll}
\Box_i\varphi \to \Diamond_i\varphi & \text{if } AD(i),\\
\Box_i\varphi \to \varphi & \text{if } AT(i),\\
\Box_i\varphi \to \Box_j\varphi & \text{if } AI(i,j),\\
\varphi \to \Box_i\Diamond_j\varphi & \text{if } AB(i,j),\\
\Box_i\varphi \to \Box_j\Box_k\varphi & \text{if } A4(i,j,k),\\
\Diamond_i\varphi \to \Box_j\Diamond_k\varphi & \text{if } A5(i,j,k).
\end{array}$$

It can be shown that the above axioms correspond to the following frame restrictions in the sense that by adding some of the axioms to the system $K_{(m)}$ we obtain an axiomatization system which is sound and complete with respect to the class of admissible interpretations that satisfy the corresponding frame restrictions.

Axiom	Corresponding condition
$\Box_i\varphi \to \Diamond_i\varphi$	$\forall u\ \exists v\ R_i(u,v)$
$\Box_i\varphi \to \varphi$	$\forall u\ R_i(u,u)$
$\Box_i\varphi \to \Box_j\varphi$	$R_j \subseteq R_i$
$\varphi \to \Box_i\Diamond_j\varphi$	$\forall u, v\ (R_i(u,v) \to R_j(v,u))$
$\Box_i\varphi \to \Box_j\Box_k\varphi$	$\forall u, v, w\ (R_j(u,v) \wedge R_k(v,w) \to R_i(u,w))$
$\Diamond_i\varphi \to \Box_j\Diamond_k\varphi$	$\forall u, v, w\ (R_i(u,v) \wedge R_j(u,w) \to R_k(w,v))$

For a *BSMM* logic L , we define the set of L-frame restrictions to be the set of the frame restrictions corresponding to the tuples of the relations *AD*, *AT*, *AI*, *AB*, *A*4, *A*5.

We sometimes use *BSMM* also to denote an arbitrary logic belonging to the *BSMM* class.

2.3. Multimodal logics of belief

To reflect properties of belief, one can extend $K_{(m)}$ with some of the following axioms:

Name	Schema	Meaning
(D)	$\Box_i\varphi \to \neg\Box_i\neg\varphi$	Belief is consistent
(I)	$\Box_i\varphi \to \Box_j\varphi$ if $i > j$	Subscript indicates degree of belief
(4)	$\Box_i\varphi \to \Box_i\Box_i\varphi$	Belief satisfies positive introspection
(4_s)	$\Box_i\varphi \to \Box_j\Box_i\varphi$	Belief satisfies strong positive introspection
(5)	$\neg\Box_i\varphi \to \Box_i\neg\Box_i\varphi$	Belief satisfies negative introspection
(5_s)	$\neg\Box_i\varphi \to \Box_j\neg\Box_i\varphi$	Belief satisfies strong negative introspection

The following systems are intended for reasoning about multidegree belief:

$$\begin{aligned} KDI4_s &= K_{(m)} + (D) + (I) + (4_s), \\ KDI4 &= K_{(m)} + (D) + (I) + (4), \\ KDI4_s5 &= K_{(m)} + (D) + (I) + (4_s) + (5), \\ KDI45 &= K_{(m)} + (D) + (I) + (4) + (5). \end{aligned}$$

In the above systems, the axiom (I) gives $\Box_i\varphi$ the meaning "φ is believed up to degree i", and $\Diamond_i\varphi$ can be read as "it is possible weakly at degree i that φ". The axioms (5) are controversial as they are quite strong. For this reason, we consider also $KDI4$ and $KDI4_s$. Note that the axiom (5_s) is derivable in $KDI4_s5$.

For multiagent systems, we use subscripts beside $\Box$ and $\Diamond$ to denote agents and assume that $\Box_i\varphi$ stands for "agent i believes that φ is true" and $\Diamond_i\varphi$ stands for "φ is considered possible by agent i". For distributed systems of belief we can use the logic system

$$KD4_s5_s = K_{(m)} + (D) + (4_s) + (5_s).$$

In this system, agents have full access to belief bases of each other. They are "friends" in a united system. In another kind of multiagent system, agents are "opponents" and they play against each other. Each one of the agents may want to simulate epistemic states of the others. To write a program for an agent, one may need to use modal operators of the other agents. A suitable logic for this problem is

$$KD45_{(m)} = K_{(m)} + (D) + (4) + (5).$$

We use a subscript in $KD45_{(m)}$ to distinguish the logic from the monomodal logic $KD45$, while there is not such a need for the other considered multimodal logics.

To capture common belief of a group of agents, one can extend the logic $KD45_{(m)}$ with modal operators for groups of agents and some additional axioms. Suppose that there are n agents and $m = 2^n - 1$. Let g be an one-to-one function that maps every natural number less than or equal to m to a nonempty subset of $\{1, \ldots, n\}$. Suppose that an index $1 \leqslant i \leqslant m$ stands for the group of agents whose indices form the set $g(i)$. We can adopt the axioms (D), (4), and additionally $(I_g) : \Box_i\varphi \to \Box_j\varphi$ if $g(i) \supset g(j)$ (i.e. i indicates a group that contains the group identified by j), and $(5_a) : \neg\Box_i\varphi \to \Box_i\neg\Box_i\varphi$ if $g(i)$ is a singleton (i.e. i stands for an agent). Thus, for reasoning about belief and common belief, we can use

$$KD4I_g5_a = K_{(m)} + (D) + (4) + (I_g) + (5_a).$$

This logic is different in the nature from the well-known multimodal logic of common knowledge. It also differs from the modal logic with mutual belief [2].

The given axioms correspond to the following frame restrictions:

Axiom	Corresponding condition
(D)	$\forall u\, \exists v\; R_i(u, v)$
(I)	$R_j \subseteq R_i$ if $i > j$
(I_g)	$R_j \subseteq R_i$ if $g(i) \supseteq g(j)$
(4)	$\forall u, v, w\; (R_i(u, v) \wedge R_i(v, w) \to R_i(u, w))$
(4_s)	$\forall u, v, w\; (R_j(u, v) \wedge R_i(v, w) \to R_i(u, w))$
(5)	$\forall u, v, w\; (R_i(u, v) \wedge R_i(u, w) \to R_i(w, v))$
(5_s)	$\forall u, v, w\; (R_j(u, v) \wedge R_i(u, w) \to R_i(v, w))$
(5_a)	as for (5) if $g(i)$ is a singleton

For further reading on modal logics, we refer the reader to [14,21,22,12].

2.4. Ordering Kripke models

A formula is in *negation normal form* if it does not contain the connective $\to$ and in which each negation occurs immediately before a classical atom. Every formula can be transformed to its equivalent negation normal form in the usual way. A formula is called *positive* if its negation normal form does not contain negation. A formula is called *negative* if its negation is a positive formula.

Definition 2.9. A model M is said to be *less than* or *equal to* N, write $M \leqslant N$, if for any positive ground formula φ, if M satisfies φ then N also satisfies φ.

The relation $\leqslant$ in the above definition is a pre-order. [3]

Definition 2.10. Let $M = \langle D, W, \tau, R_1, \ldots, R_m, \pi\rangle$ and $N = \langle D', W', \tau', R'_1, \ldots, R'_m, \pi'\rangle$ be Kripke models. We say that M is *less than or equal to N w.r.t. a binary relation* $r \subseteq W \times W'$, and write $M \leqslant_r N$, if the following conditions hold:

1. $r(\tau, \tau')$.
2. $\forall x, x', y \;\; R_i(x, y) \wedge r(x, x') \to \exists y' \; R'_i(x', y') \wedge r(y, y')$, for all $1 \leqslant i \leqslant m$.
3. $\forall x, x', y' \;\; R'_i(x', y') \wedge r(x, x') \to \exists y \; R_i(x, y) \wedge r(y, y')$, for all $1 \leqslant i \leqslant m$.
4. For any $x \in W$ and $x' \in W'$ such that $r(x, x')$, and for any ground classical atom E, if $M, x \vDash E$ then $N, x' \vDash E$.

In the above definition, the first three conditions state that r is a bisimulation of the frames of M and N. Intuitively, $r(x, x')$ states that the world x is less than or equal to x'.

Lemma 2.1. *If* $M \leqslant_r N$ *then* $M \leqslant N$.

This lemma can be proved by induction on the length of φ that, if φ is a ground formula and $M, w \vDash \varphi$ then $N, w \vDash \varphi$.

2.5. Substitution and unification

We include this subsection in order to make the paper self-contained (to a certain extent).

A *substitution* is a finite set $\theta = \{x_1/t_1, \ldots, x_k/t_k\}$, where $x_1, \ldots, x_k$ are different variables, $t_1, \ldots, t_k$ are terms, and $t_i \neq x_i$ for all $1 \leqslant i \leqslant k$. By ε we denote the *empty substitution*.

An *expression* is either a term or a formula without quantifiers, and a *simple expression* is either a term or an atom.

Let $\theta = \{x_1/t_1, \ldots, x_k/t_k\}$ be a substitution and φ be an expression. Then $\varphi\theta$, the *instance* of φ by θ, is the expression obtained from φ by simultaneously replacing all occurrences of the variable x_i in φ by the term t_i, for $1 \leqslant i \leqslant k$.

Let $\theta = \{x_1/t_1, \ldots, x_k/t_k\}$ and $\delta = \{y_1/s_1, \ldots, y_h/s_h\}$ be substitutions. Then the *composition* $\theta\delta$ of θ and δ is the substitution obtained from the set $\{x_1/(t_1\delta), \ldots, x_k/(t_k\delta), y_1/s_1, \ldots, y_h/s_h\}$ by deleting any binding $x_i/(t_i\delta)$ for which $x_i = (t_i\delta)$ and deleting any binding y_j/s_j for which $y_j \in \{x_1, \ldots, x_k\}$.

If θ and δ are substitutions such that $\theta\delta = \delta\theta = \varepsilon$, then we call them *renaming substitutions*.

We say that an expression φ is a *variant* of an expression ψ if there exist substitutions θ and γ such that $\varphi = \psi\theta$ and $\psi = \varphi\gamma$.

A substitution θ is *more general* than a substitution δ if there exists a substitution γ such that $\delta = \theta\gamma$. Note that according to our definition, θ is more general than itself.

Let Γ be a set of simple expressions. A substitution θ is called a *unifier* for Γ if $\Gamma\theta$ is a singleton. If $\Gamma\theta = \{\varphi\}$ then we say that θ unifies Γ (into φ). A unifier θ for Γ is called an mgu for Γ if θ is more general than every unifier of Γ.

There is an effective algorithm, called the *unification algorithm*, for checking whether a set Γ of simple expressions is unifiable (i.e. has a unifier) and computing an mgu for Γ if Γ is unifiable (see, e.g. [26]).

3. Positive multimodal logic programs

In [31], we presented a logic programming language called MProlog for monomodal logics. In this section, we extend this language for multimodal logics, using the same name for the new one. The defined language is as expressive as the general Horn fragment in the considered multimodal logics. For L being one of the multimodal logics of

[3] I.e. a reflexive and transitive binary relation.

belief, we adopt some restrictions on MProlog to obtain L-MProlog. The restrictions do not reduce expressiveness of the language and are acceptable from the practical point of view.

A *modality* is a (possibly empty) sequence of modal operators. A *universal modality* is a modality that contains only universal modal operators. We use $\triangle$ to denote a modality and $\boxdot$ to denote a universal modality. Similarly as in classical logic programming, we use a clausal form $\boxdot(\varphi \leftarrow \psi_1, \ldots, \psi_n)$ to denote the formula $\forall(\boxdot(\varphi \vee \neg\psi_1 \ldots \vee \neg\psi_n))$. We use E to denote a classical atom.

Definition 3.1. A *program clause* is a formula of the form $\boxdot(A \leftarrow B_1, \ldots, B_n)$, where $n \geqslant 0$ and A, B_1, …, B_n are formulas of the form E, $\Box_i E$, or $\Diamond_i E$ with E being a classical atom. $\boxdot$ is called the *modal context*, A the *head*, and $B_1, \ldots, B_n$ the *body* of the program clause.

Definition 3.2. An *MProlog program* is a finite set of program clauses.

Definition 3.3. An *MProlog goal atom* is a formula of the form $\boxdot E$ or $\boxdot\Diamond_i E$, where E is a classical atom. An *MProlog query* is a formula of the form $\exists(\alpha_1 \wedge \cdots \wedge \alpha_k)$, where α_1, …, α_k are MProlog goal atoms. An *MProlog goal* is the negation of an MProlog query, written in the form $\leftarrow \alpha_1, \ldots, \alpha_k$. We denote the *empty goal* (also called the *empty clause*) by $\Diamond$.

If P is an MProlog program, $Q = \exists(\alpha_1 \wedge \cdots \wedge \alpha_k)$ is an MProlog query and $G = \leftarrow \alpha_1, \ldots, \alpha_k$ is the corresponding goal, then $P \vDash_L Q$ iff $P \cup \{G\}$ is L-unsatisfiable. For the proof of this statement, just note that $G = \forall(\neg(\alpha_1 \wedge \cdots \wedge \alpha_k))$.

When the base logic is intended for reasoning about multidegree belief, it has little sense to write a program clause in the form $\Box_i\Box_j\varphi$ or a goal in the form $\leftarrow \Box_i\Box_j E$ or $\leftarrow \Box_i\Diamond_j E$. Besides, in the logics $KDI4_s5$ and $KD4_s5_s$ we have the tautology $\nabla\nabla'\varphi \equiv \nabla'\varphi$, where ∇ and ∇' denote modal operators. For these reasons, we introduce some restrictions for MProlog programs and goals in these logics.

Definition 3.4. For $L \in \{KDI4_s, KDI4, KDI4_s5, KDI45, KD4_s5_s\}$, an MProlog program is called an *L-MProlog program* if its program clauses have modal contexts with length 0 or 1, an MProlog goal is called an *L-MProlog goal* if its modal depth is 0 or 1. (Recall that the modal depth of φ is the maximal nesting depth of modal operators occurring in φ.)

In the logic $KD45_{(m)}$, we have the tautologies $\Box_i\Box_i\varphi \equiv \Box_i\varphi$ and $\Box_i\Diamond_i\varphi \equiv \Diamond_i\varphi$. In $KD4I_g5_a$, these two equivalences hold for the case when $g(i)$ is a singleton. So, we introduce restrictions for MProlog programs and goals in $KD45_{(m)}$ and $KD4I_g5_a$.

Definition 3.5. An MProlog program is called a $KD45_{(m)}$-*MProlog program* if the modal contexts of its program clauses do not contain subsequences of the form $\Box_i\Box_i$. An MProlog goal is called a $KD45_{(m)}$-*MProlog goal* if each of its goal atoms $\triangle E$ satisfies the condition that $\triangle$ does not contain subsequences of the form $\Box_i\Box_i$ or $\Box_i\Diamond_i$. $KD4I_g5_a$-*MProlog* programs and goals are defined similarly with the condition that $g(i)$ is a singleton.

For L not mentioned in the two above definitions, assume that no restriction is adopted for the form of L-MProlog programs and goals. In the following, we define an extension of MProlog called *eMProlog* in the same way as in [31]. It stands for the general modal Horn fragment.

Definition 3.6. A formula φ without quantifiers is called a *non-negative modal Horn formula* (*without quantifiers*) if one of the following conditions holds:
- φ is a classical atom;
- $\varphi = \psi \leftarrow \zeta$, where ψ is a non-negative modal Horn formula and ζ is a positive formula in negation normal form;
- $\varphi = \Box_i\psi$ or $\varphi = \Diamond_i\psi$ or $\varphi = \psi \wedge \zeta$, where ψ and ζ are non-negative modal Horn formulas.

Definition 3.7. An *eMProlog program* is a finite set of formulas of the form $\forall(\varphi)$, where φ is a non-negative modal Horn formula without quantifiers. An *eMProlog query* is a formula of the form $\exists(\varphi)$, where φ is a positive formula without quantifiers. An *eMProlog goal* is the negation of an eMProlog query.

We now define answers and correct answers.

Definition 3.8. Let P be an MProlog (resp. eMProlog) program and G an MProlog (resp. eMProlog) goal. An *answer* θ for $P \cup \{G\}$ is a substitution for variables of G (i.e. if $x_1, \ldots, x_n$ are all variables of G, then $\theta = \{x_{i_1}/t_1, \ldots, x_{i_k}/t_k\}$ for some $1 \leqslant i_1 < \cdots < i_k \leqslant n$ and some terms $t_1, \ldots, t_k$).

Definition 3.9. Let L be a multimodal logic, P an MProlog (resp. eMProlog) program, $Q = \exists(\varphi)$ an MProlog (resp. eMProlog) query and G the corresponding goal (i.e. $G = \neg Q$). Let θ be an answer for $P \cup \{G\}$. We say that θ is a *correct answer* in L for $P \cup \{G\}$ if $P \vDash_L \forall(\varphi\,\theta)$.

The following proposition states that MProlog and L-MProlog, where L is one of the considered multimodal logics, have the same expressiveness as eMProlog.

Proposition 3.1. *Let L be a BSMM logic. For any eMProlog program P and any eMProlog goal G, there exist an MProlog program P′ and an MProlog goal G′ such that*

- *Every correct answer in L for* $P \cup \{G\}$ *is a correct answer in L for* $P' \cup \{G'\}$ *and vice versa.*
- *If* $L \in \{KDI4_s, KDI4, KDI4_s5, KDI45, KD4_s5_s, KD45_{(m)}, KD4I_g5_a\}$, *then P′ is an L-MProlog program and G′ is an L-MProlog goal.*
- *P′ and G′ can be obtained from P and G in polynomial time.*

See [32] for the proof of this proposition.

4. Examples of application of modal logic programming

In this section, we present three examples demonstrating the usefulness of modal logic programming. The first example involves reasoning about multidegree belief, the second one involves distributed systems of belief, and the third one formalizes the wise men puzzle. Other examples can be found, e.g. in the work by Baldoni et al. [10].

Example 4.1. Assume that there are 5 degrees of belief. Consider the following program P_{mdb}:

$$\begin{aligned}
\varphi_1 &= \Box_4 good_in_maths(x) \leftarrow maths_teacher(x)\\
\varphi_2 &= \Box_5(\Box_i good_in_maths(x) \leftarrow \Box_i mathematician(x))\\
\varphi_3 &= \Box_3(\Diamond_i good_in_maths(x) \leftarrow maths_student(x))\\
\varphi_4 &= \Box_3(\Diamond_i good_in_physics(x) \leftarrow physics_student(x))\\
\varphi_5 &= \Box_2(\Diamond_2 good_in_maths(x) \leftarrow good_in_physics(x))\\
\varphi_6 &= maths_teacher(John) \leftarrow\\
\varphi_7 &= \Box_2 mathematician(Tom) \leftarrow\\
\varphi_8 &= \Box_5 maths_student(Peter) \leftarrow\\
\varphi_9 &= \Box_5 physics_student(Mike) \leftarrow .
\end{aligned}$$

The index i in the above rules can take any value from the range 1–5. Let the base logic be $KDI4_s5$. For the goal $\leftarrow \Box_4 good\_in\_maths(x)$, we have the correct answer $\{x = John\}$. For the goal $\leftarrow \Box_2 good\_in\_maths(x)$, we have the additional correct answer $\{x = Tom\}$. For the goal $\leftarrow \Diamond_1 good\_in\_maths(x)$, we have three correct answers $\{x = John\}$, $\{x = Tom\}$, and $\{x = Peter\}$.

Example 4.2. Let us consider the situation when a company has some branches and a central database. Each of the branches can access and update the database, and suppose that the company wants to distinguish data and knowledge coming from different branches. Also assume that data coming from branches can contain noises and statements expressed by a branch may not be highly recognized by other branches. This means that data and statements expressed by branches are treated as "belief" rather than "knowledge". In this case, we can use the multimodal logic $KD4_s5_s$, where each modal index represents a branch of the company, also called an *agent*. Recall that in this logic each agent has full access to the belief bases of the other agents. Data put by agent i are of the form $\Box_i E$ (agent i believes in E) or $\Diamond_i E$ (agent i considers that E is possible). A statement expressed by agent i is a clause of the form $\Box_i(A \leftarrow B_1, \ldots, B_n)$, where A is an atom of the form E, $\Box_i E$, or $\Diamond_i E$, and $B_1, \ldots, B_n$ are simple modal atoms that may contain modal operators of the other agents. For communicating with normal users, the central database may contain rules with the empty modal context, i.e. in the form $E \leftarrow B_1, \ldots, B_n$, which hide sources of information. As a concrete example, consider the following program/database P_{ddb} in $KD4_s5_s$:

agent 1:
$\varphi_1 = \Box_1 likes(Jan, cola) \leftarrow$
$\varphi_2 = \Box_1 likes(Piotr, pepsi) \leftarrow$
$\varphi_3 = \Box_1(\Diamond_1 likes(x, cola) \leftarrow likes(x, pepsi))$
$\varphi_4 = \Box_1(\Diamond_1 likes(x, pepsi) \leftarrow likes(x, cola))$

agent 2:
$\varphi_5 = \Box_2 likes(Jan, pepsi) \leftarrow$
$\varphi_6 = \Box_2 likes(Piotr, cola) \leftarrow$
$\varphi_7 = \Box_2 likes(Piotr, beer) \leftarrow$
$\varphi_8 = \Box_2(likes(x, cola) \leftarrow likes(x, pepsi))$
$\varphi_9 = \Box_2(likes(x, pepsi) \leftarrow likes(x, cola))$

agent 3:
$\varphi_{10} = \Box_3 likes(Jan, cola) \leftarrow$
$\varphi_{11} = \Diamond_3 likes(Piotr, pepsi) \leftarrow$
$\varphi_{12} = \Diamond_3 likes(Piotr, beer) \leftarrow$
$\varphi_{13} = \Box_3(very\_much\_likes(x, y) \leftarrow likes(x, y), \Box_1 likes(x, y), \Box_2 likes(x, y))$

agent communicating with users:
$\varphi_{14} = very\_much\_likes(x, y) \leftarrow \Box_3 very\_much\_likes(x, y)$
$\varphi_{15} = likes(x, y) \leftarrow \Diamond_3 very\_much\_likes(x, y)$
$\varphi_{16} = possibly\_likes(x, y) \leftarrow \Diamond_i likes(x, y).$

The modal index i in φ_{16} can take value 1, 2, or 3. Let the base logic be $KD4_s5_s$. For the goal $\leftarrow very\_much\_likes(x, y)$, we have the unique correct answer $\{x/Jan, y/cola\}$. For the goal $\leftarrow likes(x, y)$, we have two correct answers $\{x/Jan, y/cola\}$ and $\{x/Piotr, y/pepsi\}$. For the goal $\leftarrow possibly\_likes(x, y)$, we have five correct answers.

Example 4.3. The wise men puzzle is a famous benchmark introduced by McCarthy [27] for *AI*. It can be stated as follows (cf. [23]). A king wishes to know whether his three advisors (A, B, C) are as wise as they claim to be. Three chairs are lined up, all facing the same direction, with one behind the other. The wise men are instructed to sit down in the order A, B, C. Each of the men can see the backs of the men sitting before them (e.g. C can see A and B). The king informs the wise men that he has three cards, all of which are either black or white, at least one of which is white. He places one card, face up, behind each of the three wise men, explaining that each wise man must determine the color of his own card. Each wise man must announce the color of his own card as soon as he knows what it is. All know that this will happen. The room is silent; then, after a while, wise man A says “My card is white!”.

The wise men puzzle has been previously studied in a number of works (e.g. [27,23,19,16,13,5,38,11,9]). Our formalization of the wise men puzzle given below uses $KD4I_g5_a$-MProlog. It is elegant due to the clear semantics of common belief. For clarity, instead of numeric indices we use a, b, c, ab, ac, bc, abc with the meaning that $g(a) = \{a\}$, $g(b) = \{b\}$, $g(c) = \{c\}$, ..., and $g(abc) = \{a, b, c\}$. The program consists of the following clauses:

% If Y sits behind X then X’s card is white if Y considers this as possible.
$\Box_{abc}\,(white(a) \leftarrow \Diamond_b\, white(a))$
$\Box_{abc}\,(white(a) \leftarrow \Diamond_c\, white(a))$
$\Box_{abc}\,(white(b) \leftarrow \Diamond_c\, white(b))$
% The following clauses are “dual” to the above ones.
$\Box_{abc}\,(\Box_b\, black(a) \leftarrow black(a))$
$\Box_{abc}\,(\Box_c\, black(a) \leftarrow black(a))$
$\Box_{abc}\,(\Box_c\, black(b) \leftarrow black(b))$
% At least one of the wise men has a white card.
$\Box_{abc}\,(white(a) \leftarrow black(b), black(c))$
$\Box_{abc}\,(white(b) \leftarrow black(c), black(a))$
$\Box_{abc}\,(white(c) \leftarrow black(a), black(b))$

% Each of B and C does not know the color of his own card.
% In particular, each of the men considers that it is possible that his own card is black.

$\Box_{abc}\Diamond_b\, black(b)$
$\Box_{abc}\Diamond_c\, black(c).$

The goal is $\leftarrow \Box_a white(a)$, i.e. whether wise man A believes that his card is white.
See [37] for more details on this example.

5. A framework for multimodal logic programming

As mentioned earlier, there are three standard semantics for classical definite logic programs: the least model semantics, the fixpoint semantics and the SLD-resolution calculus (a procedural semantics). See Minker's work [29] for a survey and the works by Lloyd [26] and Apt [3] for foundations of classical logic programming. In this section, we give a framework for developing such mentioned semantics for L-MProlog programs. The base logic L is required to be a normal multimodal logic such that the set of L-frame restrictions consists of $\forall x\, \exists y\, R_i(x, y)$ (seriality), for all $1 \leqslant i \leqslant m$, and some classical first-order Horn clauses.

The restriction of seriality is to guarantee the existence of least models of MProlog programs. [4] Consider, for example, the following program in the nonserial modal logic K (i.e. $K_{(m)}$ with $m = 1$):

$$\Box p \leftarrow$$
$$q \leftarrow \Diamond p$$
$$s \leftarrow \Box r.$$

If there exists a world accessible from the actual world then $\Box p$ implies $\Diamond p$, which then implies q. If there does not then $\Box r$ holds and implies s. The program is thus "nondeterministic" because the accessibility relation is not serial, and consequently, it does not have any least K-model. Apart from the least model semantics, seriality is needed for our fixpoint semantics and SLD-resolution calculi for MProlog, because they are based on the assumption that $\Diamond_i$ is an "instance" of $\Box_i$.

In this section, we prove the main results using certain lemmas and theorems, which are strongly dependent on L and left as "expected". For a specific logic L, lemmas and theorems with that remark need to be proved to guarantee correctness of the main theorems w.r.t. that logic.

Our framework for developing semantics of MProlog programs is designed to be modular in the sense that it can be instantiated for different modal logics with a few details and proofs. In fact, we are able to specify all the three mentioned semantics for MProlog programs in any of the mentioned multimodal logics using only one small table that is based on the framework. Furthermore, we need to prove only "expected" lemmas and theorems for a concrete instantiation of the framework, while several important proofs given in this section remain unchanged. The "expected" lemmas point out a way for constructing a correct schema for semantics of MProlog. For modularity, proofs of "expected" lemmas and theorems that are strongly dependent on a specific logic are not presented in this section but put into a section concerning that logic (Section 6 for *BSMM* and Section 7 for $KDI4_s5$).

5.1. Labeled modal operators and notations

In classical logic programming, the direct consequence operator T_P acts on sets of ground atoms. It computes "direct" consequences of the input set using the program clauses of P. The operator is monotonic and continuous and has the least fixpoint, which is a set of atoms forming the least Herbrand model of P. In modal logic programming, to obtain

[4] In [30], we proved that every positive propositional modal logic program has a least L-model in any serial modal logic $L \in \{KD, T, KDB, B, KD4, S4, KD5, KD45, S5\}$ and can be "characterized" by two minimal L-models if L is one of the almost serial modal logics $KB, K5, K45, KB5$. On the other hand, there exist positive propositional modal logic programs that cannot be "characterized" by a finitely bounded number of models in the nonserial modal logic K, and there exists a positive propositional modal logic program that cannot be "characterized" by a finite number of models in the nonserial modal logic $K4$ (see [30]).

a similar result we first have to decide what is the domain of the direct consequence operator $T_{L,P}$. Naturally, we still want it to be the class of sets of *atoms*. But what is an *atom* in this case? When applying $T_{L,P}$, if we obtain some atom of the form $\triangle\Diamond_i E$ (where $\triangle$ is a modality and E is a classical atom), then to simplify the task we label the modal operator $\Diamond_i$. Labeling allows us to address the chosen world(s) in which this particular E must hold. A natural way is to label $\Diamond_i$ by E to obtain $\langle E\rangle_i$. Thus, an output/input of $T_{L,P}$ consists of atoms of the form $\triangle E$, where $\triangle$ is a sequence of modal operators of the form $\Box_i$ or $\langle F\rangle_i$, with E, F being ground classical atoms.

On the other hand, when dealing with SLD-derivation, we cannot change a goal $\leftarrow \Diamond_i(A \wedge B)$ to $\leftarrow \Diamond_i A, \Diamond_i B$. But if we label the operator $\Diamond_i$, let us say by X, to fix it, then we can safely change $\leftarrow \langle X\rangle_i(A \wedge B)$ to $\leftarrow \langle X\rangle_i A, \langle X\rangle_i B$.

We will use the following notations:

- $\top$: the *truth* symbol, with the usual semantics [5] ;
- E, F : classical atoms (which may contain variables) or $\top$;
- X, Y, Z : variables for classical atoms or $\top$, called *atom variables*;
- $\langle E\rangle_i$, $\langle X\rangle_i$: $\Diamond_i$ labeled by E or X;
- ∇ : $\Box_i$, $\Diamond_i$, $\langle E\rangle_i$, or $\langle X\rangle_i$, called a modal operator;
- $\triangle$: a (possibly empty) sequence of modal operators, called a *modality*;
- $\boxdot$: a *universal modality* (i.e. a modality containing only universal modal operators);
- A, B : formulas of the form E or ∇E, called *simple atoms*;
- α, β : formulas of the form $\triangle E$, called *atoms*;
- φ, ψ : (*labeled*) *formulas* (i.e. formulas that may contain $\langle E\rangle_i$ and $\langle X\rangle_i$).

We use subscripts beside ∇ to indicate modal indices in the same way as for $\Box$ and $\Diamond$. To distinguish a number of modal operators we use superscripts, e.g. ∇', $\nabla^{(i)}$, $\nabla^{(i')}$.

A *ground formula* is redefined to be a formula with no variables and no atom variables. A modal operator is said to be *ground* if it is $\Box_i$, $\Diamond_i$, or $\langle E\rangle_i$ with E being $\top$ or a ground classical atom. A *ground modality* is a modality that contains only ground modal operators. A *labeled modal operator* is a modal operator of the form $\langle E\rangle_i$ or $\langle X\rangle_i$.

We redefine also substitutions in order to deal with atom variables and labeled formulas. The other definitions involving with substitution and unification change accordingly in the usual way.

Definition 5.1. A *substitution* θ is a (finite or infinite) set of the form $\{x_1/t_1, x_2/t_2, \ldots, X_1/E_1, X_2/E_2, \ldots, Y_1/Z_1, Y_2/Z_2, \ldots\}$, where $x_1, x_2, \ldots$ are distinct variables, $t_1, t_2, \ldots$ are terms, $X_1, X_2, \ldots, Y_1, Y_2, \ldots$ are distinct atom variables, and for any element v/s of the set, s is distinct from v. The set $\{x_1, x_2, \ldots, X_1, X_2, \ldots, Y_1, Y_2, \ldots\}$ is called the *domain* of θ and denoted by $Dom(\theta)$. A substitution θ is said to be *ground* if the set $\{Y_1, Y_2, \ldots\}$ is empty, $t_1, t_2, \ldots$ are ground terms, and $E_1, E_2, \ldots$ are ground classical atoms.

Denote $EdgeLabels = \{\langle E\rangle_i \mid E \in \mathcal{B} \cup \{\top\}$ and $1 \leqslant i \leqslant m\}$, where $\mathcal{B}$ is the Herbrand base (i.e. the set of all ground classical atoms). The semantics of $\langle E\rangle_i \in EdgeLabels$ is specified below.

Definition 5.2. Let $M = \langle D, W, \tau, R_1, \ldots, R_m, \pi\rangle$ be a Kripke model. A *$\Diamond$-realization function on M* is a partial function $\sigma : W \times EdgeLabels \to W$ such that if $\sigma(w, \langle E\rangle_i) = u$, then $R_i(w, u)$ holds and $M, u \vDash E$. Given a $\Diamond$-realization function σ, a world $w \in W$, and a ground formula φ, the satisfaction relation $M, \sigma, w \vDash \varphi$ is defined in the usual way, except that $M, \sigma, w \vDash \langle E\rangle_i\psi$ iff $\sigma(w, \langle E\rangle_i)$ is defined and $M, \sigma, \sigma(w, \langle E\rangle_i) \vDash \psi$. We write $M, \sigma \vDash \varphi$ to denote that $M, \sigma, \tau \vDash \varphi$. For a set I of ground atoms, we write $M, \sigma \vDash I$ to denote that $M, \sigma \vDash \alpha$ for all $\alpha \in I$; we write $M \vDash I$ and call M a model of I if $M, \sigma \vDash I$ for *some* σ.

Definition 5.3. Let σ and σ' be $\Diamond$-realization functions on a model M. We say that σ is an *extension* of σ' if whenever $\sigma'(w, \langle E\rangle_i)$ is defined then $\sigma(w, \langle E\rangle_i) = \sigma'(w, \langle E\rangle_i)$. We say that σ is a *maximal* $\Diamond$-realization function on M if $\sigma(w, \langle E\rangle_i)$ is defined whenever $M, w \vDash \Diamond_i E$.

[5] I.e. it is always true that $M, V, w \vDash \top$.

Atom variables in modal operators of the form $\langle X\rangle_i$ are mainly interpreted by substitutions. When a formula φ is taken to be semantically considered, all modal operators $\langle X\rangle_i$ in φ are treated as[6] $\langle\top\rangle_i$, which is formalized by the following definition.

Definition 5.4. Given a Kripke model M, a $\Diamond$-realization function σ, and a labeled formula φ without quantifiers, we write $M, \sigma \models \forall_c(\varphi)$ to denote that for any substitution θ which substitutes every variable by a ground term and does not substitute atom variables, $M, \sigma \models \varphi\,\theta\,\delta_\top$, where $\delta_\top = \{X/\top \mid X \text{ is an atom variable}\}$. By $M \models \forall_c(\varphi)$ we denote $M, \sigma \models \forall_c(\varphi)$ for *some* σ.

If Γ is a set of formulas without labeled modal operators, I is a set of ground atoms, and φ is a formula without quantifiers, then the relations $\Gamma \models_L I$ and $\Gamma \models_L \forall_c(\varphi)$ are interpreted as usual.

The quantifier $\forall_c$ is introduced because $\Diamond$-realization functions are defined using Herbrand base and we do not want to restrict only to Herbrand models. Suppose that there are enough constant symbols, for example, infinitely many. Then, because a *BSMM* logic L has a complete axiomatization, for Γ being a finite formula set and φ a formula—both without labeled modal operators, $\Gamma \models_L \forall(\varphi)$ iff $\Gamma \models_L \forall_c(\varphi)$.

5.2. Model generators

As mentioned earlier, we will define the direct consequence operator $T_{L,P}$ for an MProlog program P so that an output/input of $T_{L,P}$ consists of atoms of the form $\triangle E$, where $\triangle$ is a sequence of modal operators of the form $\Box_i$ or $\langle F\rangle_i$, with E, F being ground classical atoms. For the reason that the least fixpoint of $T_{L,P}$ should represent a least L-model of P, we call inputs/outputs of $T_{L,P}$ *model generators*.

Definition 5.5. A *model generator* is a set of ground atoms not containing $\Diamond_i$, $\langle\top\rangle_i$, $\top$.

Because an atom in L may be reducible to some more compact form, for each specific logic L we will define *L-normal form of modalities*. It is possible that no restrictions on L-normal form of modalities are adopted.

Definition 5.6. A modality $\triangle$ is in *L-normal labeled form* if it is in L-normal form and does not contain modal operators of the form $\Diamond_i$ or $\langle\top\rangle_i$. An atom is in *L-normal* (*labeled*) *form* if it is of the form $\triangle E$ with $\triangle$ in L-normal (labeled) form. (Recall that E denotes a classical atom or $\top$.) An atom is in *almost L-normal labeled form* if it is of the form $\triangle A$ with $\triangle$ in L-normal labeled form. (Recall that A denotes a simple atom of the form E or ∇E, where ∇ is a modal operator possibly not labeled.)

As an example, define that a modality is in $KDI4_s5$-normal form if its length is 0 or 1. (This is justified by the $KDI4_s5$-tautology $\nabla\nabla'\varphi \equiv \nabla'\varphi$ with ∇ and ∇' being unlabeled modal operators.) In this example, let $F \neq \top$. Then the modalities $\Box_i$ and $\langle F\rangle_i$ are in $KDI4_s5$-normal labeled form, while $\Box_i\Box_j$, $\Diamond_i$, $\langle\top\rangle_i$ are not. Atoms E, $\Box_i E$, $\langle F\rangle_i E$ are in $KDI4_s5$-normal labeled form, while $\Box_i\Box_j E$, $\Diamond_i E$, $\langle\top\rangle_i E$ are not. Atoms E, $\Box_i E$, $\Diamond_i E$, $\Box_i\Box_j E$, $\Box_i\Diamond_j E$, $\langle F\rangle_i E$ are in almost $KDI4_s5$-normal labeled form, while $\Diamond_i\Box_j E$ and $\Box_i\Box_j\Box_k E$ are not.

Definition 5.7. An *L-normal model generator* is a set of ground atoms in L-normal form and not containing $\Diamond_i$, $\langle\top\rangle_i$, $\top$.

An L-normal model generator I is expected to represent an L-model. This specific model is called the *standard L-model* of I. It should contain only (positive) information that come from I. This means that the standard L-model of I should be a least L-model of I.

Given an L-normal model generator I, we can construct a least L-model for it by building an L-model graph realizing I (cf. [30]). Formulas of the form $\Box_i\alpha$ are realized in the usual way; a formula of the form $\langle E\rangle_i\alpha$ is realized at a world

[6] Atom variables appear only in *goal* bodies (see Definition 3.3). In the negation of a goal (i.e. a query) they are existentially quantified. Hence it is sufficient to choose some concrete values for them. Furthermore, as we will see, the modal operator $\langle\top\rangle_i$ plays the role of $\Box_i$; and if X remains at the end as an unsubstituted atom variable then $\langle X\rangle_i$ intuitively also plays the role of $\Box_i$.

w by connecting w to a world identified by $w\langle E\rangle_i$ via R_i and adding α to that world. To guarantee the constructed model graph to be the smallest, each new world is connected via each R_i to an empty world at the time of its creation. Sometimes, the accessibility relations are extended to satisfy all of the L-frame restrictions.

We want to give here a more declarative definition of the standard L-model of an L-normal model generator I. The part specific to L is extracted into Ext_L and $Serial_L$, where $Ext_L(I)$ is an L-normal model generator extending I, and $Serial_L$ is a set of atoms of the form $\boxdot\langle\top\rangle_i\top$. The standard L-model of I is then defined using $Ext_L(I)$ and $Serial_L$ in a unified way, almost independently from L . The set $Serial_L$ is intended to guarantee that, for every world w and $1 \leqslant i \leqslant m$, w will be connected to a world which is "less than or equal to" every world accessible from w via R_i.

Definition 5.8. Define $Serial_L = \{\boxdot\langle\top\rangle_i\top \mid 1 \leqslant i \leqslant m$ and $\boxdot\langle\top\rangle_i$ is in L−normal form$\}$.

A *forward rule* is a schema of the form $\alpha \to \beta$, while a *backward rule* is a schema of the form $\alpha \leftarrow \beta$. (Recall that we use α and β to denote atoms, i.e. formulas of the form $\triangle E$.) A rule can be accompanied with some conditions specifying when the rule can be applied. We use forward rules to specify the operators Ext_L and Sat_L (needed for defining fixpoint semantics) and use backward rules as meta-clauses when dealing with SLD-resolution calculi. In practice, conditions for applying a backward rule can be attached to the body of the rule, and in general, a backward rule can be of the form $(\alpha \leftarrow \varphi, \beta, \psi)$ with φ and ψ being conjunctions of classical atoms. In this work, we just define that a backward rule is of the form $\alpha \leftarrow \beta$.

Definition 5.9. The *operator Ext_L* is specified by a finite set of forward rules. Given an L-normal model generator I, $Ext_L(I)$ is the least extension of I that contains all ground atoms in L-normal labeled form that are derivable from some atom of I using the rules specifying Ext_L.

Note that $Ext_L(I)$ is an L-normal model generator if so is I.

As an example, for $L = KDI4_s5$, the operator Ext_L is specified by the only rule: $\Box_i E \to \Box_j E$ if $i > j$; and $Ext_L(\{\Box_2 E\}) = \{\Box_2 E, \Box_1 E\}$.

Definition 5.10. Let I be an L-normal model generator. The *standard L-model* of I is defined as follows. Let $W' = EdgeLabels^*$ (i.e. the set of all finite sequences of elements of $\{\langle E\rangle_i \mid E \in \mathcal{B} \cup \{\top\}$ and $1 \leqslant i \leqslant m\}$, where $\mathcal{B}$ is the Herbrand base), $\tau = \varepsilon$, $H(\tau) = Ext_L(I) \cup Serial_L$. Let $R'_i \subseteq W' \times W'$ and $H(u)$, for $u \in W'$, $u \neq \tau$, be the least sets such that

- if $\langle E\rangle_i\alpha \in H(w)$, then $R'_i(w, w\langle E\rangle_i)$ holds and $\{E, \alpha\} \subseteq H(w\langle E\rangle_i)$;
- if $\Box_i\alpha \in H(w)$ and $R'_i(w, w\langle E\rangle_i)$ holds, then $\alpha \in H(w\langle E\rangle_i)$.

Let R_i, for $1 \leqslant i \leqslant m$, be the least extension of R'_i such that $\{R_i \mid 1 \leqslant i \leqslant m\}$ satisfies all the L-frame restrictions except seriality (which is cared by $Serial_L$).[7] Let W be W' without worlds not accessible directly nor indirectly from τ via the accessibility relations R_i. We call the model graph $\langle W, \tau, R_1, \ldots, R_m, H\rangle$ the *standard L-model graph* of I, and its corresponding model M the *standard L-model* of I. $\{R'_i \mid 1 \leqslant i \leqslant m\}$ is called the *skeleton* of M. By the *standard $\Diamond$-realization function on M* we call the $\Diamond$-realization function σ defined as follows: if $R'_i(w, w\langle E\rangle_i)$ holds then $\sigma(w, \langle E\rangle_i) = w\langle E\rangle_i$, else $\sigma(w, \langle E\rangle_i)$ is undefined.

Example 5.1. Let us give an example for the above construction. Consider the L-normal model generator $I = \{\langle p(a)\rangle_1\, p(a), \Box_1 q(a), \Box_2 q(b)\}$ in $L = KDI4_s5$, with $m = 2$ (recall that m is the maximal modal index). We have $Ext_L(I) = I \cup \{\Box_1 q(b)\}$ (due to the rule $\Box_i E \to \Box_j E$ if $i > j$) and $Serial_L = \{\langle\top\rangle_1\top, \langle\top\rangle_2\top\}$. The standard L-model of I is specified as follows:

- $W = \{\tau, \langle p(a)\rangle_1, \langle\top\rangle_1, \langle\top\rangle_2\}$ is the set of possible worlds.
- τ is the actual world.
- $R_1 = W \times W_1$ and $R_2 = W \times W_2$ are the accessibility relations, where $W_1 = \{\langle p(a)\rangle_1, \langle\top\rangle_1\}$ and $W_2 = W_1 \cup \{\langle\top\rangle_2\}$.
- The world τ is empty; the world $\langle p(a)\rangle_1$ contains $p(a), q(a), q(b)$; the world $\langle\top\rangle_1$ contains $\top, q(a), q(b)$; the world $\langle\top\rangle_2$ contains $\top$ and $q(b)$.

[7] The least extension exists due to the assumption that all L-frame restrictions not concerning seriality are classical first-order Horn clauses.

Definition 5.11. If a modality $\triangle$ is obtainable from $\triangle'$ by replacing some (possibly zero) ∇_i by $\Box_i$ then we call $\triangle$ a *$\Box$-lifting form* of $\triangle'$. If $\triangle$ is a $\Box$-lifting form of $\triangle'$ then we call an atom $\triangle\alpha$ a *$\Box$-lifting form* of $\triangle'\alpha$. For example, $\Box_1\langle p(a)\rangle_1\Box_2 q(b)$ is a $\Box$-lifting form of $\langle X\rangle_1\langle p(a)\rangle_1\Diamond_2 q(b)$.

The following lemma will be used to prove, among others, Lemma 5.2.

Lemma 5.1. *Let it I be an L-normal model generator and $M = \langle W, \tau, R_1, \ldots, R_m, H\rangle$ the standard L-model graph of I. Let $w = \langle E_1\rangle_{i_1} \ldots \langle E_k\rangle_{i_k}$ be a world of M and $\triangle = w$ be a modality. Then for α not containing $\top$, $\alpha \in H(w)$ iff there exists a $\Box$-lifting form $\triangle'$ of $\triangle$ such that $\triangle'\alpha \in Ext_L(I)$.*

This lemma can be proved by induction on the length of w in a straightforward way.

The expected results concerning model generators are:

Expected Lemma 5.2. *Let I be an L-normal model generator, M the standard L-model of I, and σ the standard $\Diamond$-realization function on M. Then M is an L-model and $M, \sigma \models I$.*

This lemma states that the definition of standard *L*-models is well formed (i.e. the standard *L*-model of an *L*-normal model generator *I* is really an *L*-model of *I*). This lemma will be used (only) to prove the following expected theorem. Its proof is given for $L = BSMM$ in Section 6 and for $L = KDI4_s5$ in Section 7.1.

Expected Theorem 5.3. *The standard L-model of an L-normal model generator I is a least L-model of I.*

This theorem is proved for $L = BSMM$ in Section 6 and for $L = KDI4_s5$ in Section 7.1.

(We have a difficulty of calling the above assertions. Other ways are to call them *axioms* or a *lemma/theorem to be proved*. The name "axiom" is not very suitable here, because one would not say "proof of an axiom".)

5.3. Fixpoint semantics

We now return to the direct consequence operator $T_{L,P}$. Given an *L*-normal model generator *I*, how can $T_{L,P}(I)$ be defined? Basing on the axioms of L , I is first extended to the *L-saturation* of *I* denoted by $Sat_L(I)$, which is a set of atoms. Next, *L-instances of program clauses* of *P* are *applied* to the atoms of $Sat_L(I)$. This is done by the operator $T_{0L,P}$. The set $T_{0L,P}(Sat_L(I))$ is a model generator but not necessary in *L*-normal form. Finally, the *normalization operator* NF_L converts $T_{0L,P}(Sat_L(I))$ to an *L*-normal model generator. $T_{L,P}(I)$ is defined as $NF_L(T_{0L,P}(Sat_L(I)))$.

We will define a pre-order $\preccurlyeq_L$ between modal operators for each specific logic *L* to decide whether a given modality is an *L*-instance of another one. We require that $\Diamond_i \preccurlyeq_L \langle E\rangle_i \preccurlyeq_L \Box_i$, $\Diamond_i \preccurlyeq_L \langle X\rangle_i \preccurlyeq_L \Box_i$, and if $\nabla \preccurlyeq_L \langle E\rangle_i$ and $\nabla \neq \langle E\rangle_i$ then $\nabla \preccurlyeq_L \langle X\rangle_i$. Note that the condition of seriality plays an essential role here. As an example, we have the following definition.

Definition 5.12. For *L* being one of the considered multimodal logics, define $\preccurlyeq_L$ to be the least reflexive and transitive relation between modal operators such that

- $\Diamond_i \preccurlyeq_L \langle E\rangle_i \preccurlyeq_L \Box_i$ and $\Diamond_i \preccurlyeq_L \langle X\rangle_i \preccurlyeq_L \Box_i$,
- $\Box_i \preccurlyeq_L \Box_j$ and $\Diamond_j \preccurlyeq_L \Diamond_i$ if $L \in \{KDI4_s, KDI4, KDI4_s5, KDI45\}$ and $i \leqslant j$,
- $\Box_i \preccurlyeq_L \Box_j$ and $\Diamond_j \preccurlyeq_L \Diamond_i$ if $L = KD4I_g5_a$ and $g(i) \subseteq g(j)$.

Definition 5.13. An atom $\nabla^{(1)} \ldots \nabla^{(n)}\alpha$ is called an *L-instance* of an atom $\nabla^{(1')} \ldots \nabla^{(n')}\alpha'$ if there exists a substitution θ such that $\alpha = \alpha'\theta$ and, for $1 \leqslant i \leqslant n$, $\nabla^{(i)} \preccurlyeq_L \nabla^{(i')}\theta$ (treating $\nabla^{(i')}$ as an expression). A modality $\triangle$ is called an *L-instance* of $\triangle'$ if $\triangle E$ is an *L*-instance of $\triangle' E$ for some ground classical atom *E*. In that case, we also say that $\triangle'$ is *equal to* or *more general in L than* $\triangle$ (hereby we define a *pre-order between modalities*).

For example, an atom $\Box_1\Diamond_2 E$ is a $KDI4_s5$-instance of $\Box_2\langle F\rangle_1 E$, and the modality $\Box_1\Diamond_2$ is a $KDI4_s5$-instance of $\Box_2\langle F\rangle_1$.

Expected Lemma 5.4. *If $\Box_{i_1} \ldots \Box_{i_h}$ is a $\Box$-lifting form of a modality $\triangle$ in L-normal labeled form and $\triangle$ is an L-instance of $\boxdot$, then $\boxdot\varphi \models_L \Box_{i_1} \ldots \Box_{i_h}\varphi$ for any formula φ without labeled modal operators.*

This lemma clearly holds for the considered multimodal logics with $\preccurlyeq_L$ defined in Definition 5.12.

Definition 5.14. Let $\boxdot$ be a universal modality in L-normal form and $\boxdot'$ a modal context of an L-MProlog program clause. We say that $\boxdot$ is an *L-context instance* of $\boxdot'$ if $\boxdot'\varphi \to \boxdot\varphi$ is L-valid (for every φ).

Observe that if the problem of checking validity in the *propositional* version of L is decidable then the problem of checking whether $\boxdot$ is an L-context instance of $\boxdot'$ is also decidable. For all of the multimodal logics of belief considered in this work, these two problems are decidable and the latter is much simpler. [8]

Definition 5.15. Let φ and φ' be program clauses with empty modal context, $\boxdot$ a universal modality in L-normal form, and $\boxdot'$ a modal context of an L-MProlog program clause. We say that $\boxdot\varphi$ is an *L-instance* of (a program clause) $\boxdot'\varphi'$ if $\boxdot$ is an L-context instance of $\boxdot'$ and there exists a substitution θ such that $\varphi = \varphi'\theta$.

For example, $\boxdot$ is a $KDI4_s5$-context instance of $\boxdot'$ iff $\boxdot$ is a $KDI4_s5$-instance of $\boxdot'$ (i.e. either $\boxdot$ and $\boxdot'$ are empty or $\boxdot = \Box_i$, $\boxdot' = \Box_j$, and $i \leqslant j$), and we have that $\Box_1(p(a) \leftarrow q(a))$ is a $KDI4_s5$-instance of $\Box_2(p(x) \leftarrow q(x))$.

We now give definitions concerning Sat_L, $T_{0L,P}$, and NF_L.

Definition 5.16. The *saturation operator Sat_L* is specified by a finite set of forward rules. Given an L-normal model generator I, $Sat_L(I)$ is the least extension of I that contains all ground atoms in almost L-normal labeled form that are derivable from some atom in I using the rules specifying Sat_L.

As an example, for $L = KDI4_s5$, the operator Sat_L is specified by three rules: (a) $\Box_i E \to \Box_j E$ if $i > j$, (b) $\Box_i E \to \Box_m \Box_i E$, (c) $\langle F \rangle_i E \to \Box_m \Diamond_i E$; and we have $Sat_L(\{\Box_2 p(a)\}) = \{\Box_2 p(a), \Box_1 p(a), \Box_m\Box_2 p(a), \Box_m\Box_1 p(a)\}$. (Recall that m is the maximal modal index.)

We expect the following property of Sat_L (which is proved for $L = BSMM$ in Section 6 and for $L = KDI4_s5$ in Section 7.1).

Expected Lemma 5.5. *Let I be an L-normal model generator, M the standard L-model of I, and α a ground L-MProlog goal atom. Suppose that $M \models \alpha$. Then α is an L-instance of some atom of $Sat_L(I)$.*

When computing the least fixpoint of a modal logic program, whenever an atom of the form $\triangle\Diamond_i E$ is introduced, we "fix" the $\Diamond_i$ by replacing the atom by $\triangle\langle E \rangle_i E$. This leads to the following definition.

Definition 5.17. The *forward labeled form* of an atom α is the atom α' such that if α is of the form $\triangle\Diamond_i E$ then $\alpha' = \triangle\langle E \rangle_i E$, else $\alpha' = \alpha$.

For example, the forward labeled form of $\Diamond_1 s(a)$ is $\langle s(a) \rangle_1 s(a)$.

Definition 5.18. Let P be an L-MProlog program. The *operator $T_{0L,P}$* is defined as follows: for a set I of ground atoms in almost L-normal labeled form, $T_{0L,P}(I)$ is the least (w.r.t. $\subseteq$) model generator such that if $\boxdot(A \leftarrow B_1, \ldots, B_n)$ is a ground L-instance of some program clause of P and $\triangle$ is a maximally general [9] ground modality in L-normal labeled form such that $\triangle$ is an L-instance of $\boxdot$ and $\triangle B_i$ is an L-instance of some atom of I (for every $1 \leqslant i \leqslant n$), then the forward labeled form of $\triangle A$ belongs to $T_{0L,P}(I)$.

---

[8] Let $\boxdot$ and $\boxdot'$ be as in Definition 5.14. For $L \in \{KDI4_s, KDI4, KDI4_s5, KDI45, KD4_s5_s, KD45_{(m)}\}$ and the L-normal form of modalities defined later in Tables 2–6, $\boxdot$ is an L-context instance of $\boxdot'$ iff $\boxdot = \boxdot'$ or one of the following conditions holds:

- $L \in \{KDI4_s5, KD4_s5_s\}$ and $\boxdot$ is an L-instance of $\boxdot'$;
- $L = KDI4_s$, $\boxdot' = \Box_i$, and the last modal operator of $\boxdot$ is $\Box_j$ with $j \leqslant i$;
- $L \in \{KDI4, KDI45\}$, $\boxdot' = \Box_i$, $\boxdot$ is not empty, and every modal operator $\Box_j$ of $\boxdot$ satisfies $j \leqslant i$.

[9] W.r.t. the pre-order between modalities described earlier for L.

For example, if P consists of the only clause $\Box_2(\Diamond_1 p(x) \leftarrow q(x), r(x), \Box_1 s(x), \Diamond_2 t(x))$ and $I = \{\langle q(a)\rangle_1 q(a), \langle q(a)\rangle_1 r(a), \Box_2\Box_2 s(a), \Box_2\langle t(a)\rangle_1 t(a)\}$ and $L = KDI4_s5$, then $T_{0L,P}(I) = \{\langle q(a)\rangle_1\langle p(a)\rangle_1 p(a)\}$.

Definition 5.19. The *normalization operator* NF_L is specified by a finite set of forward rules. Given a model generator I, $NF_L(I)$ is the set of all ground atoms in L-normal labeled form that are derivable from some atom of I using the rules specifying NF_L.

We require that if I is a singleton then $NF_L(I)$ is also a singleton. If there are no conditions on L-normal form of atoms, then the set of rules specifying NF_L is empty and $NF_L(I) = I$.

As an example, for $L = KDI4_s5$, the operator NF_L is specified by the only rule: $\nabla\nabla' E \to \nabla' E$ if ∇' is of the form $\Box_i$ or $\langle E\rangle_i$; and we have $NF_L(\{\langle q(a)\rangle_1\langle p(a)\rangle_1 p(a)\}) = \{\langle p(a)\rangle_1 p(a)\}$.

Definition 5.20. Define $T_{L,P}(I) = NF_L(T_{0L,P}(Sat_L(I)))$.

Lemma 5.6. *The operator $T_{L,P}$ is monotonic and continuous, and it has the least fixpoint $T_{L,P}\uparrow\omega = \bigcup_{n=0}^{\omega} T_{L,P}\uparrow n$, where $T_{L,P}\uparrow 0 = \emptyset$, and $T_{L,P}\uparrow n = T_{L,P}(T_{L,P}\uparrow(n-1))$ for $n > 0$.*

Proof. The operator $T_{L,P}$ is monotonic and compact because Sat_L, $T_{0L,P}$ and NF_L are all increasingly monotonic and compact. It follows that $T_{L,P}$ is continuous. The second assertion of the lemma follows from the Kleen theorem. $\Box$

Notation 5.21. Denote the least fixpoint $T_{L,P}\uparrow\omega$ by $I_{L,P}$ and the standard L-model of $I_{L,P}$ by $M_{L,P}$.

Definition 5.22. Let P be an L-MProlog program. An L-normal model generator I is called an *L-model generator of P* if $T_{L,P}(I) \subseteq I$.

As a property of the least fixpoint, $I_{L,P}$ is the least (w.r.t. $\subseteq$) L-model generator of P.

Example 5.2. Consider the following program P in $L = KDI4_s5$:

$$\begin{array}{ll} \Diamond_1 s(a) \leftarrow & \Box_1(q(x) \leftarrow r(x), s(x)) \\ \Box_1(\Box_1 r(x) \leftarrow s(x)) & \Box_2(p(x) \leftarrow \Diamond_2 q(x)). \end{array}$$

The least L-model generator of P is $I_{L,P} = \{\langle s(a)\rangle_1 s(a), \Box_1 r(a), \langle s(a)\rangle_1 q(a), \Box_2 p(a), \Box_1 p(a)\}$.

We expect the following lemmas:

Expected Lemma 5.7. *If P is an L-MProlog program then $P \vDash_L I_{L,P}$.*

This lemma is proved for $L = BSMM$ in Section 6 and for $L = KDI4_s5$ in Section 7.1.

Expected Lemma 5.8. *Let P be an L-MProlog program and I an L-model generator of P. Then the standard L-model of I is an L-model of P.*

This lemma is proved for $L = BSMM$ in Section 6 and for $L = KDI4_s5$ in Section 7.1.

Using the two above lemmas and Expected Theorem 5.3, we can derive:

Theorem 5.9. *For an L-MProlog program P, $M_{L,P}$ is a least L-model of P.*

Proof. By Lemma 5.8, $M_{L,P}$ is an L-model of P. Let M be an arbitrary L-model of P. By Lemma 5.7, $M \vDash I_{L,P}$. Hence, by Theorem 5.3, $M_{L,P} \leqslant M$. Therefore $M_{L,P}$ is a least L-model of P. $\Box$

5.4. SLD-resolution

The fixpoint semantics can be viewed as a bottom-up method for computing answers. It repeatedly applies clauses of a given program P in order to compute the set $I_{L,P}$ of facts derivable in L from the program. Given an atom α from $I_{L,P}$, the process of tracing back the derivation of α in L from P is called top-down, because it reduces the atom, treated as a goal, to subgoals. A more general problem is to find answers for an L-MProlog goal w.r.t. an L-MProlog program. We study this problem using SLD-resolution.

The main work in developing an SLD-resolution calculus for L-MProlog is to specify a reverse analog of the operator $T_{L,P}$. While $T_{L,P}$ acts on model generators (with only ground atoms), the expected reverse analog of $T_{L,P}$ will act on goals (with variables). The operator $T_{L,P}$ is a composition of Sat_L, $T_{0L,P}$, and NF_L. So, we have to investigate reversion of these operators.

Definition 5.23. A *goal* is a clause of the form $\leftarrow \alpha_1, \ldots, \alpha_k$, where each α_i is an atom.

The following definition concerns reversion of the operator $T_{0L,P}$.

Definition 5.24. Let $G = \leftarrow \alpha_1, \ldots, \alpha_i, \ldots, \alpha_k$ be a goal and $\varphi = \boxdot(A \leftarrow B_1, \ldots, B_n)$ a program clause. Then G' is *derived* from G and φ in L using an mgu θ, and called an *L-resolvent* of G and φ, if the following conditions hold:

- $\alpha_i = \triangle' A'$, with $\triangle'$ in L-normal labeled form, is called the *selected atom*, and A' is called the *selected head atom*;
- $\triangle'$ is an L-instance of a universal modality $\boxdot'$ and $\boxdot'(A \leftarrow B_1, \ldots, B_n)$ is an L-instance of the program clause φ;
- θ is an mgu of A' and the forward labeled form of A;
- G' is the goal $\leftarrow (\alpha_1, \ldots, \alpha_{i-1}, \triangle' B_1, \ldots, \triangle' B_n, \alpha_{i+1}, \ldots, \alpha_k)\theta$.

For example, the unique $KDI4_s5$-resolvent of $\leftarrow \Box_1 p(x)$ and $\Box_2(p(x) \leftarrow \Diamond_2 q(x))$ is $\leftarrow \Box_1\Diamond_2 q(x)$ (here, $\boxdot = \Box_2$ and $\triangle' = \boxdot' = \Box_1$). As another example, the unique $KDI4_s5$-resolvent of $\leftarrow \langle Y\rangle_1\Box_1 r(x), \langle X\rangle_1 s(x)$ and $\Box_1(\Box_1 r(x) \leftarrow s(x))$ is $\leftarrow \langle Y\rangle_1 s(x), \langle X\rangle_1 s(x)$ (here, $\boxdot = \boxdot' = \Box_1$ and $\triangle' = \langle Y\rangle_1$).

As a reverse analog of the operator Sat_L, we provide the operator $rSat_L$.

Definition 5.25. The *operator* $rSat_L$ is specified by a finite set of backward rules. We say that $\beta = rSat_L(\alpha)$ *using an* $rSat_L$ *rule* $\alpha' \leftarrow \beta'$ if $\alpha \leftarrow \beta$ is of the form $\alpha' \leftarrow \beta'$. We write $\beta = rSat_L(\alpha)$ to denote that "$\beta = rSat_L(\alpha)$ using some $rSat_L$ rule".

We require that one of the $rSat_L$ rules is the *backward labeling rule* $\triangle\Diamond_i E \leftarrow \triangle\langle X\rangle_i E$ with X being a fresh [10] atom variable. We call $\triangle\langle X\rangle_i E$ a *backward labeled form* of $\triangle\Diamond_i E$.

Definition 5.26. Let $G = \leftarrow \alpha_1, \ldots, \alpha_i, \ldots, \alpha_k$ be a goal. If $\alpha_i' = rSat_L(\alpha_i)$ using an $rSat_L$ rule φ, then $G' = \leftarrow \alpha_1, \ldots, \alpha_{i-1}, \alpha_i', \alpha_{i+1}, \ldots, \alpha_k$ is *derived* from G and φ, and we call G' an (L-)*resolvent* of G and φ, and α_i the *selected atom* of G.

For example, resolving $\leftarrow \Box_1\Diamond_2 p(x)$ with the rule $\nabla\nabla' E \leftarrow \nabla' E$ results in $\leftarrow \Diamond_2 p(x)$, since ∇ is instantiated to $\Box_1$, and ∇' is instantiated to $\Diamond_2$.

As a reverse analog of the operator NF_L, we provide the operator rNF_L.

Definition 5.27. The *operator* rNF_L is specified by a finite set of backward rules. We say that $\beta =_\theta rNF_L(\alpha)$ *using an* rNF_L *rule* $\alpha' \leftarrow \beta'$ if θ is an mgu such that $\alpha\theta \leftarrow \beta$ is of the form $\alpha' \leftarrow \beta'$. We write $\beta =_\theta rNF_L(\alpha)$ if "$\beta =_\theta rNF_L(\alpha)$ using some rNF_L rule".

As an example, for $L = KDI4_s5$, the operator rNF_L is specified by the only rule: $\nabla E \leftarrow \langle X\rangle_j \nabla E$ if ∇ is of the form $\Box_i$ or $\langle E\rangle_i$, and X is a fresh atom variable; and we have $\langle Y\rangle_1\langle E\rangle_2 E =_\theta rNF_L(\langle X\rangle_2 E)$ with $\theta = \{X/E\}$ and Y being a fresh atom variable.

Definition 5.28. Let $G = \leftarrow \alpha_1, \ldots, \alpha_i, \ldots, \alpha_k$ be a goal. If $\alpha_i' =_\theta rNF_L(\alpha_i)$ using an rNF_L rule φ, then $G' = \leftarrow \alpha_1\theta, \ldots, \alpha_{i-1}\theta, \alpha_i', \alpha_{i+1}\theta, \ldots, \alpha_k\theta$ is *derived* from G and φ using the mgu θ, and we call G' an (L -)*resolvent* of G and φ, and α_i the *selected atom* of G.

[10] This means that *standardizing* is also needed for atom variables.

Observe that $rSat_L$ rules and rNF_L rules are similar to program clauses and the way of applying them is similar to the way of applying classical program clauses, except that we do not need mgu's for $rSat_L$ rules.

We now define SLD-derivation and SLD-refutation.

Definition 5.29. Let P be an L-MProlog program and G be a goal. An *SLD-derivation* from $P \cup \{G\}$ in L consists of a (finite or infinite) sequence $G_0 = G, G_1, \dots$ of goals, a sequence $\varphi_1, \varphi_2, \dots$ of variants of program clauses of P, $rSat_L$ rules, or rNF_L rules, and a sequence $\theta_1, \theta_2, \dots$ of mgu's such that if φ_i is a variant of a program clause or an rNF_L rule then G_i is derived from G_{i-1} and φ_i in L using θ_i, else $\theta_i = \varepsilon$ (the empty substitution) and G_i is derived from G_{i-1} and (the $rSat_L$ rule variant) φ_i.

We require that each φ_i in the above definition does not have any variable or atom variable which already appears in the derivation up to G_{i-1}. This can be achieved by subscripting variables and atom variables in G by 0 and in φ_i by i. This process of renaming variables is usually called *standardizing* the variables *apart* (see [26]). Each φ_i is called an *input clause/rule* of the derivation.

Definition 5.30. An *SLD-refutation* of $P \cup \{G\}$ in L is a finite SLD-derivation from $P \cup \{G\}$ in L which has the empty clause (denoted by $\Diamond$) as the last goal in the derivation.

Definition 5.31. Let P be an L-MProlog program and G be a goal. A *computed answer* θ in L of $P \cup \{G\}$ is the substitution obtained by restricting the composition $\theta_1 \dots \theta_n$ to the variables of G, where $\theta_1, \dots, \theta_n$ is the sequence of mgu's used in an SLD-refutation of $P \cup \{G\}$ in L .

Example 5.3. Consider the following program P and the goal $G = \leftarrow \Box_1 p(x)$ in $L = KDI4_s5$:

$$\begin{aligned}
\varphi_1 &= \Box_2(p(x) \leftarrow \Diamond_2 q(x))\\
\varphi_2 &= \Box_1(q(x) \leftarrow r(x), s(x))\\
\varphi_3 &= \Box_1(\Box_1 r(x) \leftarrow s(x))\\
\varphi_4 &= \Diamond_1 s(a) \leftarrow .
\end{aligned}$$

Assume that the operators rNF_L and $rSat_L$ are specified by the following rules:

$$\begin{array}{ll}
rNF_L: & \text{(a) } \nabla E \leftarrow \langle X\rangle_j \nabla E \;\text{ if } \nabla \text{ is of the form } \Box_i \text{ or } \langle E\rangle_i, \text{ and } X \text{ is a fresh atom variable}\\
rSat_L: & \text{(b) } \triangle\Diamond_i E \leftarrow \triangle\langle X\rangle_i E \;\text{ for } X \text{ being a fresh atom variable}\\
 & \text{(c) } \triangle\nabla_i \alpha \leftarrow \triangle\Box_j \alpha \;\text{ if } i \leqslant j\\
 & \text{(d) } \triangle\Diamond_i E \leftarrow \triangle\Diamond_j E \;\text{ if } i > j\\
 & \text{(e) } \nabla\nabla' E \leftarrow \nabla' E \;\text{ if } \nabla' \text{ is of the form } \Box_i \text{ or } \Diamond_i.
\end{array}$$

Here is an SLD-refutation of $P \cup \{G\}$ in L with computed answer $\{x/a\}$:

Goals	Input clauses/rules	mgu's
$\leftarrow \Box_1 p(x)$		
$\leftarrow \Box_1 \Diamond_2 q(x)$	φ_1	$\{x_1/x\}$
$\leftarrow \Diamond_2 q(x)$	(e)	ε
$\leftarrow \Diamond_1 q(x)$	(d)	ε
$\leftarrow \langle X\rangle_1 q(x)$	(b)	ε
$\leftarrow \langle X\rangle_1 r(x), \langle X\rangle_1 s(x)$	φ_2	$\{x_5/x\}$
$\leftarrow \Box_1 r(x), \langle X\rangle_1 s(x)$	(c)	ε
$\leftarrow \langle Y\rangle_1 \Box_1 r(x), \langle X\rangle_1 s(x)$	(a)	ε
$\leftarrow \langle Y\rangle_1 s(x), \langle X\rangle_1 s(x)$	φ_3	$\{x_8/x\}$
$\leftarrow \langle X\rangle_1 s(a)$	φ_4	$\{x/a, Y/s(a)\}$
$\Diamond$	φ_4	$\{X/s(a)\}$.

5.5. Soundness and completeness of SLD-resolution

We prove soundness and completeness of SLD-resolution for L-MProlog using certain "expected" lemmas, which are strongly dependent on concrete instantiations of the framework for L. Informally, an SLD-resolution calculus is sound if every computed answer for $P \cup \{G\}$ is a correct answer for $P \cup \{G\}$, and is complete if for every correct answer θ for $P \cup \{G\}$ there exists a computed answer γ for $P \cup \{G\}$ that is more general (in the sense that $G\theta = G\gamma\delta$ for some substitution δ).

Definition 5.32. We say that an atom β is *derivable* from α using $rSat_L$ (resp. (i) rNF_L, (ii) $rSat_L$ and rNF_L) if there exists a sequence of atoms $\alpha_0, \ldots, \alpha_k$ with $k \geqslant 0$, $\alpha_0 = \alpha$ and $\alpha_k = \beta$ such that for every $1 \leqslant i \leqslant k$, $\alpha_i = rSat_L(\alpha_{i-1})$ (resp. (i) $\alpha_i =_{\theta_i} rNF_L(\alpha_{i-1})$ for some θ_i, (ii) $\alpha_i = rSat_L(\alpha_{i-1})$ or $\alpha_i =_{\theta_i} rNF_L(\alpha_{i-1})$ for some θ_i).

The main results are proved using the following expected properties of $rSat_L$ and rNF_L:

Expected Lemma 5.10. *Let $\triangle$ and $\triangle'$ be ground modalities in L-normal labeled form. Let B be an atom of the form E, $\Diamond_i E$, or $\Box_i E$, and B' an atom of the form E, $\Diamond_j E$, $\langle X \rangle_j E$, or $\Box_j E$, where X is a fresh atom variable. Suppose that $\triangle$ is an L-instance of $\triangle'$ and B is an L-instance of B'. Then $\triangle' B'$ is derivable from $\triangle B$ using $rSat_L$.*

Expected Lemma 5.11. *Suppose that β is an atom in almost L-normal labeled form and $\alpha \in Sat_L(\{\beta\})$ or $\alpha \in NF_L(\{\beta\})$. Then there exists an atom β' and a substitution θ s.t. $\beta = \beta'\theta$, the domain of θ consists of fresh atom variables, and β' is derivable from α using $rSat_L$ and rNF_L.*

Expected Lemma 5.12. *Let $\beta = rSat_L(\alpha)$, M be an L-model, σ a $\Diamond$-realization function on M, and θ a substitution. Suppose that $M, \sigma \vDash \forall_c(\beta'\theta)$ for some $\Box$-lifting form β' of β. Then $M, \sigma \vDash \forall_c(\alpha'\theta)$ for some $\Box$-lifting form α' of α.*

Expected Lemma 5.13. *Let $\beta =_\delta rNF_L(\alpha)$, M be an L-model, σ a maximal $\Diamond$-realization function on M, and θ a substitution. Suppose that $M, \sigma \vDash \forall_c(\beta'\theta)$ for some $\Box$-lifting form β' of β. Then $M, \sigma \vDash \forall_c(\alpha'\delta\theta)$ for some $\Box$-lifting form α' of α.*

These lemmas are proved for $L = BSMM$ in Section 6 and for $L = KDI4_s5$ in Section 7.1.

5.5.1. Soundness

We first prove the following auxiliary lemma.

Lemma 5.14. *Let M be a Kripke model, σ a $\Diamond$-realization function on M, and θ a substitution. Suppose that $\triangle^{(1)}, \ldots, \triangle^{(l)}$ are $\Box$-lifting forms of $\triangle$ and $M, \sigma \vDash \forall_c((\triangle^{(1)} B_1 \wedge \cdots \wedge \triangle^{(l)} B_l)\theta)$. Then there exists the most general L-instance $\triangle'$ of $\triangle^{(1)}, \ldots, \triangle^{(l)}$, which is a $\Box$-lifting form of $\triangle$ and satisfies $M, \sigma \vDash \forall_c((\triangle' B_1 \wedge \cdots \wedge \triangle' B_l)\theta)$.*

Proof. Let $h = |\triangle|$ (the number of modal operators in $\triangle$). For $1 \leqslant j \leqslant l$ and $1 \leqslant k \leqslant h$, let $\nabla^{(j,k)}$ be the modal operator at position k of $\triangle^{(j)}$, and $\nabla^{(k)}$ the modal operator at position k of $\triangle$. Let i_k be the modal index (i.e. subscript) of the modal operator $\nabla^{(k)}$. If $\nabla^{(j,k)} = \Box_{i_k}$ for all $1 \leqslant j \leqslant l$, then let $\nabla^{(k')} = \Box_{i_k}$, else let $\nabla^{(k')} = \nabla^{(k)}$. Let $\triangle' = \nabla^{(1')} \ldots \nabla^{(h')}$. Clearly, $\triangle'$ is the most general L-instance of $\triangle^{(1)}, \ldots, \triangle^{(l)}$ and is a $\Box$-lifting form of $\triangle$.

Because that for $1 \leqslant j \leqslant l$, $\triangle^{(j)}$ is a $\Box$-lifting form of $\triangle'$ and $M, \sigma \vDash \forall_c((\triangle^{(j)} B_j)\theta)$, it can be proved by induction on k that $M, \sigma \vDash \forall_c((\nabla^{(1')} \ldots \nabla^{(k')} \top)\theta)$, for $1 \leqslant k \leqslant h$. It follows that $M, \sigma \vDash \forall_c((\triangle' \top)\theta)$. Because $\triangle^{(j)}$ is a $\Box$-lifting form of $\triangle'$, for $1 \leqslant j \leqslant l$, and $M, \sigma \vDash \forall_c((\triangle^{(1)} B_1 \wedge \cdots \wedge \triangle^{(l)} B_l)\theta)$, we conclude that $M, \sigma \vDash \forall_c((\triangle' B_1 \wedge \cdots \wedge \triangle' B_l)\theta)$. $\Box$

The soundness theorem is based on the following lemma:

Lemma 5.15. *Let P be an L-MProlog program and $G = \leftarrow \alpha_1, \ldots, \alpha_k$ be a goal. Then for every computed answer θ in L for $P \cup \{G\}$ there exists a goal $G' = \leftarrow \alpha'_1, \ldots, \alpha'_k$ such that α'_i is a $\Box$-lifting form of α_i, for $1 \leqslant i \leqslant k$, and $P \vDash_L \forall_c((\alpha'_1 \wedge \cdots \wedge \alpha'_k)\theta)$.*

Proof. Let M be an arbitrary L-model of P and σ a maximal $\Diamond$-realization function on M. Let the refutation of $P \cup \{G\}$ in L consist of a sequence $G_0 = G, G_1, \ldots, G_n$ of goals, a sequence $\varphi_1, \ldots, \varphi_n$ of variants of program clauses of P, $rSat_L$ rules, or rNF_L rules, and a sequence $\theta_1, \ldots, \theta_n$ of mgu's. Let θ be the computed answer. We prove by induction on n that for every $1 \leqslant i \leqslant k$ there exists a $\Box$-lifting form α'_i of α_i such that $M, \sigma \vDash \forall_c((\alpha'_1 \wedge \cdots \wedge \alpha'_k)\theta)$.

Suppose that $n = 1$. This means that $G = \leftarrow \alpha_1$ with $\alpha_1 = \triangle' A'$, A' is the selected head atom, and the empty clause is an L-resolvent of G and some input clause $\varphi_1 = \boxdot(A \leftarrow)$. By Lemma 5.4, $P \vDash_L \forall(\Box_{i_1} \ldots \Box_{i_h} A)$, where $\Box_{i_1} \ldots \Box_{i_h}$ is a $\Box$-lifting form of $\triangle'$. If A' is of the form $\Box_i E$ or E, then $A'\theta_1 = A\theta_1$, and $P \vDash_L \forall(\Box_{i_1} \ldots \Box_{i_h} A'\theta_1)$. Suppose that $A' = \langle F \rangle_i E'$ or $A' = \langle X \rangle_i E'$. Thus $A = \Diamond_i E$. Let $A'' = \langle E \rangle_i E$ (the forward labeled form of A). We have $A'\theta_1 = A''\theta_1 = \langle E'' \rangle_i E''$ for some E''. Since $P \vDash_L \forall(\Box_{i_1} \ldots \Box_{i_h} A)$, we have $P \vDash_L \forall(\Box_{i_1} \ldots \Box_{i_h} \Diamond_i E'')$. It follows that $M, \sigma \vDash \forall_c(\Box_{i_1} \ldots \Box_{i_h} \langle E'' \rangle_i E'')$, because M is an L-model of P and σ is a maximal $\Diamond$-realization function on M. Hence $M, \sigma \vDash \forall_c(\Box_{i_1} \ldots \Box_{i_h} A'\theta_1)$. Thus, for $\alpha'_1 = \Box_{i_1} \ldots \Box_{i_h} A'$, we have $M, \sigma \vDash \forall_c(\alpha'_1\theta)$.

Next suppose that the result holds for computed answers which come from refutations of length less than n. There are the following cases: G_1 is derived from G and an $rSat_L$/rNF_L rule variant, or G_1 is an L-resolvent of G and a variant of some program clause of P. The case G_1 is derived from G and an $rSat_L$ rule variant immediately follows from the inductive assumption and Lemma 5.12.

Suppose that G_1 is derived from G and an rNF_L rule variant, α_i is the selected atom and it is replaced by $\beta =_{\theta_1} rNF_L(\alpha_i)$. We have

$$G_1 = \leftarrow \alpha_1\theta_1, \ldots, \alpha_{i-1}\theta_1, \beta, \alpha_{i+1}\theta_1 \ldots, \alpha_k\theta_1.$$

By the inductive assumption, there exist a $\Box$-lifting form α'_j of α_j, for $1 \leqslant j \leqslant k$ and $j \neq i$, and a $\Box$-lifting form β' of β such that

$$M, \sigma \vDash \forall_c((\alpha'_1\theta_1 \wedge \cdots \wedge \alpha'_{i-1}\theta_1 \wedge \beta' \wedge \alpha'_{i+1}\theta_1 \wedge \cdots \wedge \alpha'_k\theta_1)\theta_2 \ldots \theta_n).$$

We have $M, \sigma \vDash \forall_c(\beta'\theta_2 \ldots \theta_n)$. Hence, by Lemma 5.13, there exists a $\Box$-lifting form α'_i of α_i such that $M, \sigma \vDash \forall_c(\alpha'_i\theta_1\theta_2 \ldots \theta_n)$. Therefore $M, \sigma \vDash \forall_c((\alpha'_1 \wedge \cdots \wedge \alpha'_k)\theta)$.

Now suppose that G_1 is derived in L from G and an input clause $\varphi = \boxdot(A \leftarrow B_1, \ldots, B_l)$ ($l \geqslant 0$), the selected atom is $\alpha_i = \triangle' A'$, and A' is the selected head atom. We have

$$G_1 = \leftarrow (\alpha_1, \ldots, \alpha_{i-1}, \triangle' B_1, \ldots, \triangle' B_l, \alpha_{i+1}, \ldots, \alpha_k)\theta_1.$$

By the inductive assumption, there exists a goal

$$G'_1 = \leftarrow (\alpha'_1, \ldots, \alpha'_{i-1}, \triangle^{(1')} B'_1, \ldots, \triangle^{(l')} B'_l, \alpha'_{i+1}, \ldots, \alpha'_k)\theta_1$$

such that

$$M, \sigma \vDash \forall_c((\alpha'_1 \wedge \cdots \wedge \alpha'_{i-1} \wedge \triangle^{(1')} B'_1 \wedge \cdots \wedge \triangle^{(l')} B'_l \wedge \alpha'_{i+1} \wedge \cdots \wedge \alpha'_k)\theta),$$

where α'_j is a $\Box$-lifting form of α_j, for $1 \leqslant j \leqslant k$ and $j \neq i$, and $\triangle^{(j')} B'_j$ is a $\Box$-lifting form of $\triangle' B_j$ with $|\triangle^{(j')}| = |\triangle'|$, for $1 \leqslant j \leqslant l$. Let $\Box_{i_1} \ldots \Box_{i_h}$ be a $\Box$-lifting form of $\triangle'$, and $\triangle''$ be the most general L-instance of $\triangle^{(1')}, \ldots, \triangle^{(l')}$ if $l > 0$, which exists due to Lemma 5.14, and be $\Box_{i_1} \ldots \Box_{i_h}$ otherwise. By Lemma 5.14, $\triangle''$ is a $\Box$-lifting form of $\triangle'$, and $M, \sigma \vDash \forall_c((\triangle'' B'_1 \wedge \cdots \wedge \triangle'' B'_l)\theta)$ if $l > 0$. Since M is an L-model of P, by Lemma 5.4, we have $M \vDash \forall(\Box_{i_1} \ldots \Box_{i_h}(B_1 \wedge \cdots \wedge B_l \rightarrow A))$. Hence $M, \sigma \vDash \forall_c((\triangle'' A)\theta)$ (because $\Box_{i_1} \ldots \Box_{i_h}$ is a $\Box$-lifting form of $\triangle''$, B'_j is a $\Box$-lifting form of B_j, and L is a serial modal logic). Let A'' be the forward labeled form of A. Since σ is a maximal $\Diamond$-realization function on M, it follows that $M, \sigma \vDash \forall_c((\triangle'' A'')\theta)$. Since $A'\theta_1 = A''\theta_1$, by choosing $\alpha'_i = \triangle'' A'$, we have that α'_i is a $\Box$-lifting form of α_i and $M, \sigma \vDash \forall_c((\alpha'_1 \wedge \cdots \wedge \alpha'_k)\theta)$. This completes the proof. $\Box$

Theorem 5.16 (*Soundness of SLD-resolution*). *Let P be an L-MProlog program and G an L-MProlog goal. Then every computed answer in L for $P \cup \{G\}$ is a correct answer in L for $P \cup \{G\}$.*

Proof. Let $G = \leftarrow \alpha_1, \ldots, \alpha_k$, where each α_i is of the form $\boxdot E$ or $\boxdot\Diamond E$. Let θ be a computed answer in L for $P \cup \{G\}$. Since L is a serial modal logic, by Lemma 5.15, we have $P \vDash_L \forall_c((\alpha_1 \wedge \cdots \wedge \alpha_k)\theta)$. Assume that the signature contains enough constant symbols, for example, infinitely many. Then it follows that $P \vDash_L \forall((\alpha_1 \wedge \cdots \wedge \alpha_k)\theta)$. Hence θ is a correct answer in L for $P \cup \{G\}$. $\Box$

5.5.2. Completeness

We use a standard method to prove completeness of our SLD-resolution calculus (cf. [26,25]). In general, completeness of a resolution calculus is first proved for the ground version and then lifted to the case with variables. The flow of this subsection follows Lloyd [26]. The proofs of Lemmas 5.17, 5.18, 5.22 and Theorem 5.23 are very similar to the ones given for classical logic programming in Lloyd's book, but we present all of them to make the paper self-contained.

We first define unrestricted SLD-refutation and give the *mgu lemma* and the *lifting lemma*.

Definition 5.33. An *unrestricted SLD-refutation* in L is an SLD-refutation in L , except that we drop the requirement that the substitutions θ_i be most general unifiers. They are only required to be unifiers. In an unrestricted SLD-resolution, if a goal G_i is derived from G_{i-1} and an $rSat_L$ rule variant, then θ_i can be arbitrary and $G_i = G'_i\theta_i$, where G'_i is the goal derived from G_{i-1} and that $rSat_L$ rule variant in the usual way.

Lemma 5.17 (*mgu lemma*). *Let P be an L-MProlog program and G be a goal. Suppose that $P \cup \{G\}$ has an unrestricted SLD-refutation in L . Then $P \cup \{G\}$ has an SLD-refutation in L of the same length such that, if $\theta_1, \ldots, \theta_n$ are the unifiers from the unrestricted refutation and $\theta'_1, \ldots, \theta'_n$ are mgu's from the refutation, then there exists a substitution γ such that $\theta_1 \ldots \theta_n = \theta'_1 \ldots \theta'_n\gamma$.*

Proof. Let the unrestricted refutation of $P \cup \{G\}$ consist of a sequence $G_0 = G, G_1, \ldots, G_n$ of goals, a sequence $\varphi_1, \ldots, \varphi_n$ of variants of program clauses of P, $rSat_L$ rules, or rNF_L rules, and a sequence $\theta_1, \ldots, \theta_n$ of unifiers. We prove the result by induction on n.

Suppose that $n = 1$. This means that $G = \leftarrow \triangle' A'$ and the empty clause is an L-resolvent of G and the input clause $\varphi_1 = \boxdot(A \leftarrow)$, where A' is the selected head atom. Let θ'_1 be an mgu of A' and the forward labeled form of A. Then $\theta_1 = \theta'_1\gamma$ for some γ. Furthermore, $P \cup \{G\}$ has a refutation in L consisting of $G_0 = G$, $G_1 = \Diamond$ (the empty goal) with input clause φ_1 and mgu θ'_1.

Now suppose that the result holds for unrestricted refutations with length less than n. Let $G = \leftarrow \alpha_1, \ldots, \alpha_k$ and α_i be the selected atom of G.

Suppose that G_1 is derived from G and the input clause $\varphi_1 = \boxdot(A \leftarrow B_1, \ldots, B_l)$ in L , the selected atom α_i is $\triangle' A'$, where A' is the selected head atom. There exists an mgu θ'_1 for A' and the forward labeled form of A. We have $\theta_1 = \theta'_1\delta$ for some δ. Let G'_1 be the goal derived in the same way as G_1 but with θ'_1 instead of θ_1. We have $G_1 = G'_1\delta$. Then G_2 can be derived from G'_1 in the same way as from G_1 but with unifier $\delta\theta_2$ instead of θ_2. Thus $P \cup \{G\}$ has an unrestricted refutation in L consisting of $G_0 = G, G'_1, G_2, \ldots, G_n$ with unifiers $\theta'_1, \delta\theta_2, \theta_3, \ldots, \theta_n$. By the inductive assumption, $P \cup \{G'_1\}$ has a refutation in L with mgu's $\theta'_2, \ldots, \theta'_n$ such that $\delta\theta_2 \ldots \theta_n = \theta'_2 \ldots \theta'_n\gamma$, for some γ. Thus $P \cup \{G\}$ has a refutation in L consisting of $G_0 = G, G'_1, \ldots, G'_n = \Diamond$ with mgu's $\theta'_1, \theta'_2 \ldots \theta'_n$ such that $\theta_1\theta_2 \ldots \theta_n = \theta'_1\delta\theta_2 \ldots \theta_n = \theta'_1\theta'_2 \ldots \theta'_n\gamma$.

The cases when G_1 is derived from G and an $rSat_L/rNF_L$ rule variant are similar to the above case. □

Lemma 5.18 (*Lifting lemma*). *Let P be an L-MProlog program, G a goal, and θ a substitution. Suppose there exists an SLD-refutation of $P \cup \{G\theta\}$ in L . Then there exists an SLD-refutation of $P \cup \{G\}$ in L of the same length such that, if $\theta_1, \ldots, \theta_n$ are the mgu's from the refutation of $P \cup \{G\theta\}$ and $\theta'_1, \ldots, \theta'_n$ are the mgu's from the refutation of $P \cup \{G\}$, then there exists a substitution γ such that $\theta\theta_1 \ldots \theta_n = \theta'_1 \ldots \theta'_n\gamma$.*

Proof. Let the refutation of $P \cup \{G\theta\}$ consist of a sequence $G_0 = G, G_1, \ldots, G_n$ of goals, a sequence $\varphi_1, \ldots, \varphi_n$ of variants of program clauses of P, $rSat_L$ rules, or rNF_L rules, and a sequence $\theta_1, \ldots, \theta_n$ of mgu's.

Suppose that G_1 is an L-resolvent of $G\theta$ and the input clause φ_1 using θ_1. We may assume that θ does not act on any variables of φ_1. Let $\varphi_1 = \boxdot(A \leftarrow B_1, \ldots, B_l)$, $G = \leftarrow \alpha_1, \ldots, \alpha_k$, and the selected atom of $G\theta$ be $\alpha_i\theta = (\triangle' A')\theta$, where $A'\theta$ is the selected head atom. Now $\theta\theta_1$ is a unifier for A' and the forward labeled form of A. The result of resolving G and φ_1 using $\theta\theta_1$ is exactly G_1. Thus we obtain an unrestricted refutation of $P \cup \{G\}$ in L , which looks exactly like the given refutation of $P \cup \{G\theta\}$, except the original goal is different and the first unifier is $\theta\theta_1$. Now apply the mgu lemma.

The cases when G_1 is derived from G and an $rSat_L/rNF_L$ rule are similar to the above case. □

The following lemma is an essential part of the completeness proof.

Lemma 5.19. *Let P be an L-MProlog program and $\alpha \in I_{L,P}$. Then $P \cup \{\leftarrow \alpha\}$ has an SLD-refutation in L.*

Proof. We prove by induction on n that if $\alpha \in T_{L,P} \uparrow n$ then $P \cup \{\leftarrow \alpha\}$ has an SLD-refutation in L. This assertion obviously holds for $n = 0$, since $T_{L,P} \uparrow 0 = \emptyset$.

Suppose that the assertion holds for $(n-1)$ in the place of n. Let $\alpha \in T_{L,P} \uparrow n$. There exist a program clause $\varphi = \boxdot(A \leftarrow B_1, \ldots, B_k)$ of P, with $k \geqslant 0$, a substitution θ, modalities $\triangle'$ and $\boxdot'$, ground atoms $\gamma_1, \ldots, \gamma_k \in T_{L,P} \uparrow (n-1)$, and ground atoms $\beta_1, \ldots, \beta_k, \alpha'$ such that

- $\beta_i \in Sat_L(\{\gamma_i\})$, for $1 \leqslant i \leqslant k$;
- $\beta_i = \triangle^{(i)} B'_i$ and $B_i\theta$ is an L-instance of B'_i, for $1 \leqslant i \leqslant k$;
- $\boxdot'$ is an L-context instance of $\boxdot$;
- $\triangle'$ is in the L-normal labeled form and is an L-instance of $\triangle^{(1)}, \ldots, \triangle^{(k)}, \boxdot'$;
- $\alpha' = \triangle' A' \theta$, where A' is the forward labeled form of A;
- $\alpha \in NF_L(\{\alpha'\})$.

By Lemma 5.11, there exist atoms $\alpha'', \gamma'_1, \gamma'_2, \ldots, \gamma'_k$, and ground substitutions $\delta_0, \ldots, \delta_k$ with disjoint domains such that

- α'' is derivable from α using $rSat_L$ and rNF_L, and $\alpha' = \alpha''\delta_0$,
- γ'_i is derivable from β_i using $rSat_L$ and rNF_L, and $\gamma_i = \gamma'_i\delta_i$, for $1 \leqslant i \leqslant k$.

Let $\delta = \delta_1 \ldots \delta_k$ if $k > 0$, and $\delta = \varepsilon$ otherwise. By the inductive assumption, $P \cup \{\leftarrow \gamma_i\}$ has a refutation in L, for $1 \leqslant i \leqslant k$. Since $\gamma'_i\delta = \gamma'_i\delta_i = \gamma_i$, it follows that $P \cup \{\leftarrow \gamma'_i\delta\}$ has a refutation in L. Hence $P \cup \{\leftarrow (\gamma'_1, \ldots, \gamma'_k)\delta\}$ has a refutation in L, since $\gamma'_i\delta$ are ground. By the lifting lemma, $P \cup \{\leftarrow \gamma'_1, \ldots, \gamma'_k\}$ has a refutation in L. Since γ'_i is derivable from β_i using $rSat_L$ and rNF_L, it follows that $P \cup \{\leftarrow \beta_1, \ldots, \beta_k\}$ has a refutation in L.

For $1 \leqslant i \leqslant k$, if B'_i is of the form $\langle F \rangle_h E$ then let $\beta'_i = \triangle^{(i)} \langle X \rangle_h E$ and $\theta_i = \{X/F\}$, where X is a fresh atom variable; else let $\beta'_i = \beta_i$ and $\theta_i = \varepsilon$. Let $\theta' = \theta_1 \ldots \theta_k$ if $k > 0$, and $\theta' = \varepsilon$ otherwise. Since $\beta_i = \beta'_i\theta'$, $P \cup \{\leftarrow (\beta'_1, \ldots, \beta'_k)\theta'\}$ has a refutation in L. Hence, by the lifting lemma, $P \cup \{\leftarrow \beta'_1, \ldots, \beta'_k\}$ has a refutation in L. Therefore, by Lemma 5.10, $P \cup \{\leftarrow \triangle' B_1\theta, \ldots, \triangle' B_k\theta\}$ has a refutation in L. The goal $\leftarrow \triangle' B_1\theta, \ldots, \triangle' B_k\theta$ is an unrestricted L-resolvent of $\leftarrow \alpha'$ and φ. Hence, by the mgu lemma, $P \cup \{\leftarrow \alpha'\}$ has a refutation in L. This means that $P \cup \{\leftarrow \alpha''\delta_0\}$ has a refutation in L. By the lifting lemma, $P \cup \{\leftarrow \alpha''\}$ has a refutation in L. Since α'' is derivable from α using $rSat_L$ and rNF_L, we conclude that $P \cup \{\leftarrow \alpha\}$ has a refutation in L. □

Corollary 5.20. *Let P be an L-MProlog program and $\alpha \in Sat_L(I_{L,P})$. Then $P \cup \{\leftarrow \alpha\}$ has an SLD-refutation in L.*

Proof. There exists $\beta \in I_{L,P}$ such that $\alpha \in Sat_L(\{\beta\})$. By Lemma 5.11, there exist an atom β' and a substitution θ such that $\beta = \beta'\theta$ and β' is derivable from α using $rSat_L$ and rNF_L. Since $\beta \in I_{L,P}$, by Lemma 5.19, $P \cup \{\leftarrow \beta\}$ has a refutation in L. This means that $P \cup \{\leftarrow \beta'\theta\}$ has a refutation in L. By the lifting lemma, $P \cup \{\leftarrow \beta'\}$ has a refutation in L. Consequently, $P \cup \{\leftarrow \alpha\}$ has a refutation in L. □

Lemma 5.21. *Let P be an L-MProlog program and α a ground L-MProlog goal atom such that $M_{L,P} \vDash \alpha$. Then $P \cup \{\leftarrow \alpha\}$ has an SLD-refutation in L.*

Proof. By Lemma 5.5, α is an L-instance of some $\alpha' \in Sat_L(I_{L,P})$. By Corollary 5.20, $P \cup \{\leftarrow \alpha'\}$ has an SLD-refutation in L. If α' is of the form E, $\triangle\Diamond_i E$, or $\triangle\Box_i E$ then, by Lemma 5.10, $P \cup \{\leftarrow \alpha\}$ has an SLD-refutation in L. If α' is of the form $\triangle\langle F \rangle_i E$ then, by the lifting lemma, $P \cup \{\leftarrow \triangle\langle X \rangle_i E\}$ has an SLD-refutation in L, where X is a fresh atom variable. By the assumption about $\preccurlyeq_L$, α is also an L-instance of $\triangle\langle X \rangle_i E$. Hence, by Lemma 5.10, $P \cup \{\leftarrow \alpha\}$ has an SLD-refutation in L. □

For the main theorem, we need also the following auxiliary lemma.

Lemma 5.22. *Let P be an L-MProlog program and α an L-MProlog goal atom. Suppose that $\forall(\alpha)$ is a logical consequence in L of P. Then there exists an SLD-refutation of $P \cup \{\leftarrow \alpha\}$ in L with the identity substitution as the computed answer.*

Proof. Suppose α has variables $x_1, \ldots, x_n$. Let $a_1, \ldots, a_n$ be distinct constants not appearing in P and α, and let θ be the substitution $\{x_1/a_1, \ldots, x_n/a_n\}$. Then it is clear that $\alpha\theta$ is a logical consequence in L of P. By Lemma 5.8, we have

$M_{L,P} \vDash \alpha\theta$. Since $\alpha\theta$ is ground, by Lemma 5.21, $P \cup \{\leftarrow \alpha\theta\}$ has a refutation in L. Since the a_i do not appear in P or α, by replacing a_i by x_i (for $1 \leqslant i \leqslant n$) in this refutation, we obtain a refutation of $P \cup \{\leftarrow \alpha\}$ in L with the identity substitution as the computed answer. $\square$

Theorem 5.23 (*Completeness of SLD-resolution*). *Let P be an L-MProlog program and G an L-MProlog goal. For every correct answer θ in L for $P \cup \{G\}$, there exists a computed answer γ in L for $P \cup \{G\}$ such that $G\theta = G\gamma\delta$ for some substitution δ.*

Proof. Suppose G is the goal $\leftarrow \alpha_1, \ldots, \alpha_k$. Since θ is a correct answer in L for $P \cup \{G\}$, $\forall((\alpha_1 \wedge \cdots \wedge \alpha_k)\theta)$ is a logical consequence of P in L. By Lemma 5.22, there exists a refutation of $P \cup \{\leftarrow \alpha_i\theta\}$ in L such that the computed answer is the identity substitution, for $1 \leqslant i \leqslant k$. We can combine these refutations into a refutation of $P \cup \{G\theta\}$ such that the computed answer is the identity substitution. Applying the lifting lemma, we conclude that there exists a refutation of $P \cup \{G\}$ in L with computed answer γ such that $G\theta = G\gamma\delta$, for some substitution δ. $\square$

5.6. Summary

We have given a framework for developing fixpoint semantics, the least model semantics, and SLD-resolution calculi for L-MProlog programs. The base logic L is required to be a normal multimodal logic such that the L-frame restrictions consist of $\forall x\, \exists y\, R_i(x, y)$ (seriality), for $1 \leqslant i \leqslant m$, and some classical first-order Horn clauses.

Definition 5.34. By a *schema for semantics of L-MProlog* we mean a table consisting of a definition of L-normal form of modalities, a definition of $\preccurlyeq_L$, and rules specifying the operators Ext_L, Sat_L, NF_L, rNF_L, $rSat_L$. We say that such a schema is *correct* if all the expected results of this section hold for L-MProlog w.r.t. that schema.

To show correctness of a schema, we have to prove Expected Theorem 5.3 and Expected Lemmas 5.2, 5.4, 5.5, 5.7, 5.8, 5.10–5.13. Theorem 5.9 has been proved using Expected Theorem 5.3 and Expected Lemmas 5.7 and 5.8. It states that the fixpoint semantics coincides with the least model semantics. Theorems 5.16 and 5.23 about soundness and completeness of SLD-resolution for L-MProlog has been proved using Expected Lemmas 5.4, 5.5, 5.8, 5.10–5.13.

6. A schema for semantics of *BSMM*-MProlog

In this section, let L be a *BSMM* logic. In Table 1, we present a schema for semantics of *BSMM*-MProlog. The first rule specifying $rSat_L$ is a generalized version of the backward labeling rule and is dual to the first rule specifying Sat_L. The remaining rules specifying Sat_L and $rSat_L$ directly come from the axioms. This gives an impression that the schema relies on syntactic properties of the base logic. Clarity of the rules suggests a general method for translating axioms of a given modal logic into an SLD-resolution calculus for that logic.

Example 6.1. Consider the multimodal logic L specified by $m = 2$ (the number of different modal indices), $AD = \{1, 2\}$, $AT = \{1\}$, $AI = \{(2, 1)\}$, and $AB = A4 = A5 = \emptyset$. In other words, the logic is characterized by the axioms: $\Box_1\varphi \to \Diamond_1\varphi$; $\Box_2\varphi \to \Diamond_2\varphi$; $\Box_1\varphi \to \varphi$; and $\Box_2\varphi \to \Box_1\varphi$. Consider the following program P:

$$\varphi_1 = \Diamond_2 p(a) \leftarrow$$
$$\varphi_2 = \Box_2(\Box_1 q(x) \leftarrow \Diamond_2 p(x))$$
$$\varphi_3 = \Box_2(r(x) \leftarrow p(x), q(x)).$$

We have $T_{L,P} \uparrow 1 = \{\langle p(a)\rangle_2 p(a)\}$ and

$$Sat_L(T_{L,P} \uparrow 1) = \{\langle p(a)\rangle_2 p(a), \langle p(a)\rangle_2 \Diamond_1 p(a), \langle p(a)\rangle_2 \Diamond_2 p(a)\}.$$

Applying the program clause φ_2 and its L-instance $\Box_1 q(x) \leftarrow \Diamond_2 p(x)$ to $Sat_L(T_{L,P} \uparrow 1)$, we obtain $T_{L,P} \uparrow 2 = T_{L,P} \uparrow 1 \cup \{\langle p(a)\rangle_2 \Box_1 q(a), \Box_1 q(a)\}$. The set $Sat_L(T_{L,P} \uparrow 2)$ contains both $\langle p(a)\rangle_2 p(a)$ and $\langle p(a)\rangle_2 q(a)$. Hence, by applying φ_3, we have $\langle p(a)\rangle_2 r(a) \in T_{L,P} \uparrow 3$ and arrive at

$$T_{L,P} \uparrow \omega = T_{L,P} \uparrow 3 = \{\langle p(a)\rangle_2 p(a), \langle p(a)\rangle_2 \Box_1 q(a), \Box_1 q(a), \langle p(a)\rangle_2 r(a)\}.$$

Table 1
A schema for semantics of *BSMM*-MProlog

$L = BSMM$, L-MProlog		
$\preccurlyeq_L$ is defined by Definition 5.12 in Section 5.3.		
No restrictions on L-normal form of modalities No rules specifying NF_L and rNF_L		
Rules specifying Ext_L and Sat_L:	$\triangle\langle E\rangle_i\alpha \to \triangle\Diamond_i\alpha$	(1)
	$\triangle\Box_i\alpha \to \triangle\Diamond_i\alpha$	(2)
	$\triangle\Box_i\alpha \to \triangle\alpha$ if $AT(i)$	(3)
	$\triangle\alpha \to \triangle\Diamond_i\alpha$ if $AT(i)$	(4)
	$\triangle\Box_i\alpha \to \triangle\Box_j\alpha$ if $AI(i, j)$	(5)
	$\triangle\Diamond_j\alpha \to \triangle\Diamond_i\alpha$ if $AI(i, j)$	(6)
	$\triangle\alpha \to \triangle\Box_i\Diamond_j\alpha$ if $AB(i, j)$	(7)
	$\triangle\Diamond_i\Box_j\alpha \to \triangle\alpha$ if $AB(i, j)$	(8)
	$\triangle\Box_i\alpha \to \triangle\Box_j\Box_k\alpha$ if $A4(i, j, k)$	(9)
	$\triangle\Diamond_j\Diamond_k\alpha \to \triangle\Diamond_i\alpha$ if $A4(i, j, k)$	(10)
	$\triangle\Diamond_i\alpha \to \triangle\Box_j\Diamond_k\alpha$ if $A5(i, j, k)$	(11)
	$\triangle\Diamond_j\Box_k\alpha \to \triangle\Box_i\alpha$ if $A5(i, j, k)$	(12)
Rules specifying $rSat_L$:	$\triangle\Diamond_i\alpha \leftarrow \triangle\langle X\rangle_i\alpha$ where X is a fresh atom variable	(1)
	$\triangle\nabla_i\alpha \leftarrow \triangle\Box_i\alpha$	(2)
	plus a rule $\alpha \leftarrow \beta$ for each kth rule $\beta \to \alpha$ specifying Sat_L, $k \geqslant 3$, with the same accompanying condition	(3)–(12)

We give below an SLD-refutation of $P \cup \{\leftarrow \Diamond_2 r(x)\}$ in L with computed answer $\{x/a\}$.

Goals	Input clauses/rules	mgu's
$\leftarrow \Diamond_2 r(x)$		
$\leftarrow \langle X\rangle_2 r(x)$	(1): $\triangle\Diamond_i\alpha \leftarrow \triangle\langle X\rangle_i\alpha$	ε
$\leftarrow \langle X\rangle_2 p(x), \langle X\rangle_2 q(x)$	$\Box_2(r(x) \leftarrow p(x), q(x))$	$\{x_2/x\}$
$\leftarrow \langle p(a)\rangle_2 q(a)$	$\Diamond_2 p(a) \leftarrow$	$\{X/p(a), x/a\}$
$\leftarrow \langle p(a)\rangle_2 \Box_1 q(a)$	(3): $\triangle\alpha \leftarrow \triangle\Box_1\alpha$	ε
$\leftarrow \langle p(a)\rangle_2 \Diamond_2 p(a)$	$\Box_2(\Box_1 q(x) \leftarrow \Diamond_2 p(x))$	$\{x_5/a\}$
$\leftarrow \langle p(a)\rangle_2 \Diamond_1 p(a)$	(6): $\triangle\Diamond_2\alpha \leftarrow \triangle\Diamond_1\alpha$	ε
$\leftarrow \langle p(a)\rangle_2 p(a)$	(4): $\triangle\Diamond_1\alpha \leftarrow \triangle\alpha$	ε
$\Diamond$	$\Diamond_2 p(a) \leftarrow$	ε.

Theorem 6.1. *The schema given in Table* 1 *for semantics of BSMM-MProlog is correct*.

To prove this theorem we have to prove Expected Theorem 5.3 and Expected Lemmas 5.2, 5.4, 5.5, 5.7, 5.8, 5.10–5.13. To do this we need *extended L-model graphs* (defined below) and some properties of them.

Definition 6.1. Let I be a model generator. Define Ext'_L to be the operator such that $Ext'_L(I)$ is the least set of atoms extending I and closed w.r.t. the rules specifying Ext_L. (Note that we allow $Ext'_L(I)$ to contain atoms not in labeled form and have that $Ext_L(I) \subseteq Ext'_L(I)$.) The *extended L-model graph* of I is defined in the same way as the standard L-model graph of I but with $Ext'_L(I)$ in the place of $Ext_L(I)$.

Lemma 6.2. *Let I be a model generator, M the standard L-model graph of I, and M' the extended L-model graph of I. Then M' has the same frame as M, and furthermore, if $M = \langle W, \tau, R_1, \ldots, R_m, H\rangle$ and $M' = \langle W, \tau, R_1, \ldots, R_m, H'\rangle$*

then for every $w \in W$, $H(w) \subseteq H'(w)$ and $H'(w) - H(w)$ is a set of formulas containing some unlabeled existential modal operators.

The proof of this lemma is straightforward.

The following lemma is similar to Lemma 5.1 and can also be proved by induction on the length of $\triangle$ in a straightforward way.

Lemma 6.3. *Let I be a model generator and $M = \langle W, \tau, R_1, \ldots, R_m, H\rangle$ be the extended L-model graph of I. Let $w = \langle E_1\rangle_{i_1} \ldots \langle E_k\rangle_{i_k}$ be a world of M and $\triangle = w$ be a modality. Then for α (resp. A)*[11] *not containing $\top$, $\alpha \in H(w)$ (resp. $A \in H(w)$) iff there exists a $\Box$-lifting form $\triangle'$ of $\triangle$ such that $\triangle'\alpha \in Ext'_L(I)$ (resp. $\triangle' A \in Sat_L(I)$).*

We give below the main lemma concerning extended L-model graphs.

Lemma 6.4. *Let I be a model generator and $M = \langle W, \tau, R_1, \ldots, R_m, H\rangle$ be the extended L-model graph of I. Then for any w and u such that $R_i(w, u)$ holds*: (i) if $\Box_i\alpha \in H(w)$ then $\alpha \in H(u)$, (ii) if $\alpha \in H(u)$ then $\Diamond_i\alpha \in H(w)$.

Proof. Let $\{R'_j \mid 1 \leqslant j \leqslant m\}$ be the skeleton of M. We prove this lemma by induction on the number of steps needed to obtain $R_i(w, u)$ when extending $\{R'_j \mid 1 \leqslant j \leqslant m\}$ to $\{R_j \mid 1 \leqslant j \leqslant m\}$.

Consider the first assertion. Suppose that $\Box_i\alpha \in H(w)$. By Lemma 6.3, there exists a $\Box$-lifting form $\triangle$ of w such that $\triangle\Box_i\alpha \in Ext'_L(I)$. Since $R_i(w, u)$ holds, there are the following cases to consider:

- Case $u = w\langle E\rangle_i$ and $R'_i(w, w\langle E\rangle_i)$: The assertion holds by the definition of M.
- Case $AT(i)$ holds and $u = w$: Since $\triangle\Box_i\alpha \in Ext'_L(I)$, we have $\triangle\alpha \in Ext'_L(I)$, and by Lemma 6.3, $\alpha \in H(u)$.
- Case $AI(i, j)$ holds and $R_i(w, u)$ is created from $R_j(w, u)$: Since $\triangle\Box_i\alpha \in Ext'_L(I)$, we have $\triangle\Box_j\alpha \in Ext'_L(I)$, and by Lemma 6.3, $\Box_j\alpha \in H(w)$. Hence, by the inductive assumption, $\alpha \in H(u)$.
- Case $AB(j, i)$ holds and $R_i(w, u)$ is created from $R_j(u, w)$: Since $\Box_i\alpha \in H(w)$, by the inductive assumption, $\Diamond_j\Box_i\alpha \in H(u)$. By Lemma 6.3, there exists a $\Box$-lifting form $\triangle'$ of u such that $\triangle'\Diamond_j\Box_i\alpha \in Ext'_L(I)$. Thus $\triangle'\alpha \in Ext'_L(I)$. Hence, by Lemma 6.3, $\alpha \in H(u)$.
- Case $A4(i, j, k)$ holds and $R_i(w, u)$ is created from $R_j(w, v)$ and $R_k(v, u)$: Since $\triangle\Box_i\alpha \in Ext'_L(I)$, we have $\triangle\Box_j\Box_k\alpha \in Ext'_L(I)$, and by Lemma 6.3, $\Box_j\Box_k\alpha \in H(w)$. Hence, by the inductive assumption, $\Box_k\alpha \in H(v)$ and $\alpha \in H(u)$.
- Case $A5(j, k, i)$ holds and $R_i(w, u)$ is created from $R_j(v, u)$ and $R_k(v, w)$: Since $\Box_i\alpha \in H(w)$, by the inductive assumption, $\Diamond_k\Box_i\alpha \in H(v)$. Hence, by Lemma 6.3, there exists a $\Box$-lifting form $\triangle'$ of v such that $\triangle'\Diamond_k\Box_i\alpha \in Ext'_L(I)$. Hence $\triangle'\Box_j\alpha \in Ext'_L(I)$, and by Lemma 6.3, $\Box_j\alpha \in H(v)$. By the inductive assumption, it follows that $\alpha \in H(u)$.

The second assertion can be proved in a similar way (see [32]). $\Box$

To increase readability we will recall expected lemmas and theorems before giving their proofs.

Expected Lemma 5.2. *Let I be an L-normal model generator, M the standard L-model of I, and σ the standard $\Diamond$-realization function on M. Then M is an L-model and $M, \sigma \vDash I$.*

Proof. By the definition, M is an L-model. Let $M' = \langle W, \tau, R_1, \ldots, R_m, H\rangle$ be the extended L-model graph of I. It can be proved by induction on the length of α that for any $w \in W$, if $\alpha \in H(w)$, then $M', \sigma, w \vDash \alpha$. The cases when α is a classical atom or $\alpha = \langle E\rangle_i\beta$ are trivial. The case when $\alpha = \Box_i\beta$ is solved by Lemmas 6.2 and 6.4. Hence $M, \sigma \vDash I$. $\Box$

Expected Theorem 5.3. *The standard L-model of an L-normal model generator I is a least L-model of I.*

Proof. Let $M = \langle W, \tau, R_1, \ldots, R_m, H\rangle$ be the standard L-model graph of I, σ the standard $\Diamond$-realization function, and $\{R'_i \mid 1 \leqslant i \leqslant m\}$ the skeleton of the standard L-model of I. By Lemma 5.2, M is an L-model of I. Let $N =$

[11] Recall that α denotes an atom of the form $\triangle'' E$, while A denotes a simple atom of the form E or ∇E, where E is a classical atom and ∇ is a modal operator.

$\langle D, W_2, \tau_2, S_1, \ldots, S_m, \pi\rangle$ be an arbitrary L-model of I and σ_2 a $\Diamond$-realization function on N such that $N, \sigma_2 \vDash I \cup Serial_L$.

Let $r \subseteq W \times W_2$ be the least relation such that, for all w, w_2, u_2, E, i:

- $r(\tau, \tau_2)$;
- if $r(w, w_2)$ and $R'_i(w, w\langle E\rangle_i)$ hold, and $\sigma_2(w_2, \langle E\rangle_i)$ is defined, then $r(w\langle E\rangle_i, \sigma_2(w_2, \langle E\rangle_i))$;
- if $r(w, w_2)$ and $S_i(w_2, u_2)$ hold, then $r(w\langle \top\rangle_i, u_2)$.

Note that if $r(w, w_2)$ and $S_i(w_2, u_2)$ hold, then for $u = w\langle\top\rangle_i$ we have $r(u, u_2)$ and $R_i(w, u)$.

We prove that $M \leqslant_r N$. We first show that if $r(u, u_2)$ and $\alpha \in H(u)$ then $N, \sigma_2, u_2 \vDash \alpha$. We prove this by induction on the length of u. Suppose that $r(u, u_2)$ holds and $\alpha \in H(u)$. The case $u = \varepsilon$ is trivial. Let $u = w\langle E\rangle_i$ and inductively assume that the assertion holds when u is replaced by w. There are two cases:

- $u_2 = \sigma_2(w_2, \langle E\rangle_i)$, $r(w, w_2)$, and $R'_i(w, w\langle E\rangle_i)$, for some $w_2 \in W_2$; or
- $E = \top$, $r(w, w_2)$, and $S_i(w_2, u_2)$, for some $w_2 \in W_2$.

Consider the first case. Since $\alpha \in H(u)$, either $\Box_i\alpha \in H(w)$ or $\langle E\rangle_i\alpha \in H(w)$. By the inductive assumption, either $N, \sigma_2, w_2 \vDash \Box_i\alpha$ or $N, \sigma_2, w_2 \vDash \langle E\rangle_i\alpha$. Hence, $N, \sigma_2, \sigma_2(w_2, \langle E\rangle_i) \vDash \alpha$, which means that $N, \sigma_2, u_2 \vDash \alpha$.

Consider the second case. Since $\alpha \in H(u)$, it follows that $\Box_i\alpha \in H(w)$. By the inductive assumption, $N, \sigma_2, w_2 \vDash \Box_i\alpha$, and hence $N, \sigma_2, u_2 \vDash \alpha$ since $S_i(w_2, u_2)$.

We now show that if $r(w, w_2)$ and $R'_i(w, w\langle E\rangle_i)$ hold then $\sigma_2(w_2, \langle E\rangle_i)$ is defined. The case $E = \top$ is trivial. Suppose that $r(w, w_2)$ and $R'_i(w, w\langle E\rangle_i)$ hold and $E \neq \top$. Thus, there exists $\langle E\rangle_i\alpha \in H(w)$ for some α. Hence $N, \sigma_2, w_2 \vDash \langle E\rangle_i\alpha$ and $\sigma_2(w_2, \langle E\rangle_i)$ is defined. Therefore, the second condition in the above definition of r can be simplified to "if $r(w, w_2)$ and $R'_i(w, w\langle E\rangle_i)$ hold then $r(w\langle E\rangle_i, \sigma_2(w_2, \langle E\rangle_i))$".

It is straightforward to prove by induction on the number of steps needed to obtain $R_i(w, u)$ when extending $\{R'_j \mid 1 \leqslant j \leqslant m\}$ to $\{R_j \mid 1 \leqslant j \leqslant m\}$ that if $R_i(w, u)$ then: (i) if $r(w, w_2)$ then there exists u_2 such that $r(u, u_2)$ and $S_i(w_2, u_2)$; (ii) if $r(u, u_2)$ then there exists w_2 such that $r(w, w_2)$ and $S_i(w_2, u_2)$. We give here only the base case, when $u = w\langle E\rangle_i$: (i) suppose that $r(w, w_2)$ holds. We have $R'_i(w, w\langle E\rangle_i)$, hence $\sigma_2(w_2, \langle E\rangle_i)$ is defined. The assertion holds for $u_2 = \sigma_2(w_2, \langle E\rangle_i)$. (ii) Suppose that $r(u, u_2)$ holds. By the definition of r, there exists w_2 such that $r(w, w_2)$ and ($S_i(w_2, u_2)$ or $u_2 = \sigma_2(w_2, \langle E\rangle_i)$). It is clear that the assertion holds for such w_2.

We have proved that r satisfies all the conditions to guarantee $M \leqslant_r N$. This together with Lemma 5.2 implies that M is a least L-model of I. $\Box$

Expected Lemma 5.4. *If $\Box_{i_1} \ldots \Box_{i_h}$ is a $\Box$-lifting form of a modality $\triangle$ in L-normal labeled form and $\triangle$ is an L-instance of $\boxdot$, then $\boxdot\varphi \vDash_L \Box_{i_1} \ldots \Box_{i_h}\varphi$ for any formula φ without labeled modal operators.*

Proof. Just note that $\boxdot = \Box_{i_1} \ldots \Box_{i_h}$ (due to Definition 5.12 of $\preccurlyeq_L$). $\Box$

Expected Lemma 5.5. *Let I be an L-normal model generator, M the standard L-model of I, and α a ground L-MProlog goal atom. Suppose that $M \vDash \alpha$. Then α is an L-instance of some atom of $Sat_L(I)$.*

Proof. Let $M' = \langle W, \tau, R_1, \ldots, R_m, H\rangle$ be the extended L-model graph of I, $\boxdot = \Box_{i_1} \ldots \Box_{i_k}$ and $w = \langle\top\rangle_{i_1} \ldots \langle\top\rangle_{i_k}$. Suppose that α is of the form $\boxdot E$. Since $M \vDash \alpha$, by Lemma 6.2, we have $M', w \vDash E$. By Lemma 6.3, it follows that $\boxdot E \in Sat_L(I)$. Now suppose that α is of the form $\boxdot\Diamond_i E$. Since $M \vDash \alpha$, we have $M, w \vDash \Diamond_i E$, and by Lemma 6.2, $M', w \vDash \Diamond_i E$. There exists u such that $R_i(w, u)$ holds and $M', u \vDash E$. By Lemma 6.4, it follows that $\Diamond_i E \in H(w)$. Hence $\boxdot\Diamond_i E \in Sat_L(I)$ (by Lemma 6.3). $\Box$

Expected Lemma 5.7. *If P is an L-MProlog program then $P \vDash_L I_{L,P}$.*

Proof. Let $M = \langle D, W, \tau, R_1, \ldots, R_m, \pi\rangle$ be an arbitrary L-model of P and σ a *maximal* $\Diamond$-realization function on M (see Definition 5.3). It is straightforward to prove by induction on n that $M, \sigma \vDash T_{L,P} \uparrow n$. In fact, if $M, \sigma \vDash T_{L,P} \uparrow n$, then $M, \sigma \vDash Sat_L(T_{L,P} \uparrow n)$, and hence $M, \sigma \vDash T_{0L,P}(Sat_L(T_{L,P} \uparrow n))$. Since $NF_L(I) = I$ for any I, it follows that $M, \sigma \vDash T_{L,P}(T_{L,P} \uparrow n)$. Therefore $M, \sigma \vDash I_{L,P}$. $\Box$

Expected Lemma 5.8. *Let P be an L-MProlog program and I an L-model generator of P. Then the standard L-model of I is an L-model of P.*

Proof. Let I' be the least extension of I such that, if $\boxdot\varphi$ is a program clause of P, $\varphi = (A \leftarrow B_1, \ldots, B_n)$, and ψ is a ground instance of φ, then $\boxdot p_\psi \in I'$, where p_ψ is a fresh 0-ary predicate symbol. Let M and M' be the extended L-model graphs of I and I', respectively. It is easy to see that these model graphs have the same frame. Let $M = \langle W, \tau, R_1, \ldots, R_m, H\rangle$ and $M' = \langle W, \tau, R_1, \ldots, R_m, H'\rangle$. Clearly, M is an L-model. By Lemma 6.2, it suffices to show that $M \vDash P$.

Let $\boxdot\varphi$ be a program clause of P, $\varphi = (A \leftarrow B_1, \ldots, B_n)$, and ψ a ground instance of φ. By Lemmas 5.2 and 6.2, $M' \vDash \boxdot\, p_\psi$. To prove that $M \vDash P$ it is sufficient to show that for any $w \in W$, if $p_\psi \in H'(w)$ then $M, w \vDash \psi$. Suppose that $p_\psi \in H'(w)$.

Let $\triangle = w$ and $\boxdot' = \Box_{i_1} \ldots \Box_{i_k}$ be a $\Box$-lifting form of $\triangle$. By Lemma 6.3, some $\Box$-lifting form of $\triangle p_\psi$ belongs to $Sat_L(I')$. This $\Box$-lifting form must be $\boxdot' p_\psi$. Thus $\boxdot' p_\psi \in Sat_L(\{\boxdot p_\psi\})$. Hence $\boxdot p_\psi \to \boxdot' p_\psi$ is L-valid and the program clause $\boxdot'\psi$ is a ground L-instance of $\boxdot\varphi$.

Let $\psi = (A' \leftarrow B'_1, \ldots, B'_n)$ and suppose that $M, w \vDash B'_i$ for all $1 \leqslant i \leqslant n$. We need to show that $M, w \vDash A'$. For this, we first show that a $\Box$-lifting form of $\triangle B'_i$ belongs to $Sat_L(I)$ for every $1 \leqslant i \leqslant n$. Consider the following cases:

- Case B'_i is a classical atom: The assertion follows from Lemma 6.3.
- Case B'_i is of the form $\Box_j E$: Since $M, w \vDash B'_i$, it follows that $M, w\langle\top\rangle_j \vDash E$, and by Lemma 6.3, some $\Box$-lifting form of $\triangle\langle\top\rangle_j E$ belongs to $Sat_L(I)$, which means that some $\Box$-lifting form of $\triangle B'_i$ belongs to $Sat_L(I)$.
- Case B'_i is of the form $\Diamond_j E$: Since $M, w \vDash B'_i$, there exists a world u such that $R_j(w, u)$ holds and $M, u \vDash E$. By Lemma 6.4, it follows that $\Diamond_j E \in H(w)$. Hence, by Lemma 6.3, some $\Box$-lifting form of $\triangle B'_i$ belongs to $Sat_L(I)$.

Therefore, by the definition of $T_{0L,P}$, some $\Box$-lifting form α of $\triangle A''$, where A'' is the forward labeled form of A', belongs to $T_{0L,P}(Sat_L(I))$. Since $T_{0L,P}(Sat_L(I)) = T_{L,P}(I) \subseteq I$, by Lemma 5.2, we have that $M, \sigma \vDash \alpha$, where σ is the standard $\Diamond$-realization function on M. Hence $M, w \vDash A'$. Thus $M, w \vDash \psi$, which completes the proof. □

Expected Lemma 5.10. *Let $\triangle$ and $\triangle'$ be ground modalities in L-normal labeled form. Let B be an atom of the form E, $\Diamond_i E$, or $\Box_i E$, and B' an atom of the form E, $\Diamond_j E$, $\langle X\rangle_j E$, or $\Box_j E$, where X is a fresh atom variable. Suppose that $\triangle$ is an L-instance of $\triangle'$ and B is an L-instance of B'. Then $\triangle' B'$ is derivable from $\triangle B$ using $rSat_L$.*

Proof. We have that $\triangle'$ is a $\Box$-lifting form of $\triangle$, and either B' is a $\Box$-lifting form of B or B' is of the form $\langle X\rangle_j$ and B is of the form $\Diamond_j$. Hence $\triangle' B'$ is derivable from $\triangle B$ using applications of the $rSat_L$ rules $\triangle\nabla_i\alpha \leftarrow \triangle\Box_i\alpha$ and $\triangle\Diamond_i\alpha \leftarrow \triangle\langle X\rangle_i\alpha$. □

Expected Lemma 5.11. *Suppose that β is an atom in almost L-normal labeled form and $\alpha \in Sat_L(\{\beta\})$ or $\alpha \in NF_L(\{\beta\})$. Then there exists an atom β' and a substitution θ s.t. $\beta = \beta'\theta$, the domain of θ consists of fresh atom variables, and β' is derivable from α using $rSat_L$ and rNF_L.*

Proof. Note that NF_L is the identity operator and we can ignore it. If α is derived from β using Sat_L rules identified by $(i_1), \ldots, (i_k)$, then by applying the sequence of $rSat_L$ rules identified by $(i_k), \ldots, (i_1)$ to α we obtain an atom β' such that $\beta = \beta'\theta$, where θ is a substitution with domain consisting of fresh atom variables. □

Expected Lemma 5.12. *Let $\beta = rSat_L(\alpha)$, M be an L-model, σ a $\Diamond$-realization function on M, and θ a substitution. Suppose that $M, \sigma \vDash \forall_c(\beta'\theta)$ for some $\Box$-lifting form β' of β. Then $M, \sigma \vDash \forall_c(\alpha'\theta)$ for some $\Box$-lifting form α' of α.*

Proof. If the rule used to derive β from α is $\triangle\nabla_i\gamma \leftarrow \triangle\Box_i\gamma$, where γ denotes an atom, then just let $\alpha' = \beta'$. The remaining cases are similar to each other, and we consider, e.g. the case when the used rule is $\triangle\Box_j\Diamond_k\gamma \leftarrow \triangle\Diamond_i\gamma$. We have that $\alpha = \triangle\Box_j\Diamond_k\gamma$ and $\beta = \triangle\Diamond_i\gamma$. Let $\beta' = \triangle'\nabla_i\gamma'$. Since $M, \sigma \vDash \forall_c(\beta'\theta)$, we have $M, \sigma \vDash \forall_c(\triangle'\Diamond_i\gamma'\theta)$, and hence $M, \sigma \vDash \forall_c(\triangle'\Box_j\Diamond_k\gamma'\theta)$ (since $A5(i, j, k)$ holds). Choose $\alpha' = \triangle'\Box_j\Diamond_k\gamma'$. □

Expected Lemma 5.13. *Let $\beta =_\delta rNF_L(\alpha)$, M be an L-model, σ a maximal $\Diamond$-realization function on M, and θ a substitution. Suppose that $M, \sigma \vDash \forall_c(\beta'\theta)$ for some $\Box$-lifting form β' of β. Then $M, \sigma \vDash \forall_c(\alpha'\delta\theta)$ for some $\Box$-lifting form α' of α.*

Proof. This lemma is irrelevant for $L = BSMM$, because there are no rules specifying rNF_L. □

7. Programming about multidegree belief

Our SLD-resolution calculus for MProlog in *BSMM* is elegant like a Hilbert-style axiom system, but similar to using a Hilbert-style axiom system for automatic reasoning, it is not very efficient. The calculus may be too "syntactic". For more specific modal logics like the mentioned multimodal logics of belief, we want to have more efficient SLD-resolution calculi. For this aim, we look more deeply at "semantical" properties of the considered logics and use advanced techniques introduced for our framework like normalizing modalities or ordering modal operators.

To reason about multidegree belief we can use the multimodal logics $KDI4$, $KDI4_s$, $KDI4_s5$, and $KDI45$. Recall that, in these logics, $\Box_i \varphi$ stands for "φ is believed up to degree i" and $\Diamond_i \varphi$ stands for "it is possible weakly at degree i that φ". In this section, we present a schema for semantics of $KDI4_s5$-MProlog and prove its correctness. Schemata for semantics of MProlog in $KDI4$, $KDI4_s$, and $KDI45$ are presented in the Appendix, and proofs of their correctness are given in [32].

7.1. A schema for semantics of $KDI4_s5$-MProlog

In this subsection, let L denote the logic $KDI4_s5$. It can be checked that a connected frame $\langle W, \tau, R_1, \ldots, R_m\rangle$ is a $KDI4_s5$-frame iff there are nonempty subsets of worlds $W_1 \subseteq \cdots \subseteq W_m$ such that $W = \{\tau\} \cup W_m$ and $R_i = W \times W_i$, for $1 \leqslant i \leqslant m$. (Recall that m is the maximal modal index; and we use E to denote a classical atom, A to denote a simple atom of the form E or ∇E, where ∇ is a modal operator, and α to denote an atom of the form $\triangle E$.)

In Section 5 we have given several small examples involving with $KDI4_s5$. In Table 2, we present a full schema for semantics of $KDI4_s5$-MProlog. L-normal form of modalities and the rules (2)–(5) and (9) in that schema are justified by the L-tautology $\nabla\varphi \equiv \nabla'\nabla\varphi$ with ∇ and ∇' being unlabeled modal operators. The rule (1) follows from the axiom (I), the rule (7) is based on the axioms (D) and (I), and the rule (8) follows from the reverse of the axiom (I).

The schema given in Table 2 is formulated so that it can use the proofs given in Section 5. However, the rules (6)–(8) of Table 2 can be simplified by deleting the occurrences of $\triangle$ and replacing α by E without violating soundness and completeness of SLD-resolution. Furthermore, the rule (7) can be deleted if: (a) the condition of the rule (5) that ∇ is of the form $\Box_i$ or $\langle E\rangle_i$ is deleted, (b) when resolving a goal with an input clause, we relax the condition that mgu θ unifies the selected head atom A' with the forward labeled form A'' of the head of the input clause, but only require that θ is a most general substitution such that $A'\theta$ and $A''\theta$ have the same classical atom and $A'\theta$ is an L-instance of $A''\theta$. It can be shown that every SLD-refutation in the original calculus can be simulated in the new calculus by another one with a more general computed answer, and vice versa. This means that the new SLD-resolution calculus is sound and complete, provided that so is the original calculus.

Example 7.1. Reconsider the MProlog program P_{mdb} given in Example 4.1. To increase readability, we recall some clauses of P_{mdb}:

$$\varphi_5 = \Box_2(\Diamond_2 good\_in\_maths(x) \leftarrow good\_in\_physics(x))$$
$$\varphi_9 = \Box_5 physics\_student(Mike) \leftarrow .$$

Here is an SLD-refutation of $P_{mdb} \cup \{\leftarrow \Diamond_2 good\_in\_maths(x)\}$ in $KDI4_s5$:

Goals	Input clauses	mgu's, constraints
$\leftarrow \Diamond_2 good\_in\_maths(x)$		
$\leftarrow \langle X\rangle_2 good\_in\_maths(x)$	(6)	ε
$\leftarrow \langle Y\rangle_j \langle good\_in\_maths(x)\rangle_2 good\_in\_maths(x)$	(5)	$\{X/good\_in\_maths(x)\}$
$\leftarrow \langle Y\rangle_j good\_in\_physics(x)$	φ_5	$\{x_3/x\}$, $j \leqslant 2$
$\leftarrow \Box_j good\_in\_physics(x)$	(7)	ε, $j \leqslant 2$
$\Diamond$	φ_9	$\{x/Mike\}$.

The computed answer is $\{x/Mike\}$. In the above refutation, j can take value 1 or 2. In another work, we have implemented MProlog as an additional module to Prolog, and constraints as goal atoms. With that module, we can also consider, for example, the goals $\leftarrow \Box_i good\_in\_maths(x)$ and $\leftarrow \Diamond_i good\_in\_maths(x)$.

Table 2
A schema for semantics of $KDI4_s5$-MProlog

$L = KDI4_s5$, L-MProlog		
$\preccurlyeq_L$ is defined by Definition 5.12 in Section 5.3.		
A modality is in L-normal form if its length $\leqslant 1$.		
Rules specifying		
Ext_L	$\Box_i E \to \Box_j E$ if $i > j$	(1)
Sat_L	The rules specifying Ext_L plus	
	$\Box_i E \to \Box_m \Box_i E$	(2)
	$\langle F\rangle_i E \to \Box_m \Diamond_i E$	(3)
NF_L	$\nabla\nabla' E \to \nabla' E$ if ∇' is of the form $\Box_i$ or $\langle E\rangle_i$	(4)
rNF_L	$\nabla E \leftarrow \langle X\rangle_j \nabla E$ if ∇ is of the form $\Box_i$ or $\langle E\rangle_i$ and X is a fresh atom variable	(5)
$rSat_L$	$\triangle\Diamond_i E \leftarrow \triangle\langle X\rangle_i E$ for X being a fresh atom variable	(6)
	$\triangle\nabla_i \alpha \leftarrow \triangle\Box_j \alpha$ if $i \leqslant j$	(7)
	$\triangle\Diamond_i E \leftarrow \triangle\Diamond_j E$ if $i > j$	(8)
	$\nabla\nabla' E \leftarrow \nabla' E$ if ∇' is of the form $\Box_i$ or $\Diamond_i$	(9)

Theorem 7.1. *The schema given in Table 2 for semantics of $KDI4_s5$-MProlog is correct.*

To prove this theorem we have to prove Expected Theorem 5.3 and Expected Lemmas 5.2, 5.4, 5.5, 5.7, 5.8, 5.10–5.13. To increase readability we will recall expected lemmas and theorems before giving their proofs.

Expected Lemma 5.2. *Let I be an L-normal model generator, M the standard L-model of I, and σ the standard $\Diamond$-realization function on M. Then M is an L-model and $M, \sigma \vDash I$.*

Proof. By the definition, M is an L-model. Let $\{R'_i \mid 1 \leqslant i \leqslant m\}$ be the skeleton of M. We prove by induction on the length of α that for any $w \in W$, if $\alpha \in H(w)$, then $M, \sigma, w \vDash \alpha$. The cases when α is a classical atom or $\alpha = \langle E\rangle_i F$ (and $w = \tau$) are trivial. Consider the remaining case when $\alpha = \Box_i E$ and $w = \tau$. Let u be a world such that $R_i(\tau, u)$ holds. We show that $E \in H(u)$. Since $R_i(\tau, u)$, u must be of the form $\langle F\rangle_j$ for some F and $j \leqslant i$. Since $\Box_i E \in H(\tau)$, by the definition of Ext_L, we have $\Box_j E \in H(\tau)$, and hence $E \in H(u)$. $\Box$

Expected Theorem 5.3. *The standard L-model of an L-normal model generator I is a least L-model of I.*

Proof. Let $M = \langle W, \tau, R_1, \ldots, R_m, H\rangle$ be the standard L-model graph of I, σ the standard $\Diamond$-realization function and $\{R'_i \mid 1 \leqslant i \leqslant m\}$ the skeleton of the standard L-model of I. By Lemma 5.2, M is an L-model of I. Let $N = \langle D, W_2, \tau_2, S_1, \ldots, S_m, \pi\rangle$ be an arbitrary L-model of I and σ_2 a $\Diamond$-realization function on N such that $N, \sigma_2 \vDash I \cup Serial_L$.

We first show that if $R'_i(\tau, \langle E\rangle_i)$ holds then $\sigma_2(\tau_2, \langle E\rangle_i)$ is defined. The case $E = \top$ is trivial. Suppose that $R'_i(\tau, \langle E\rangle_i)$ holds and $E \neq \top$. Thus, there exists $\langle E\rangle_i \alpha \in H(\tau)$ for some α. Hence $N, \sigma_2, \tau_2 \vDash \langle E\rangle_i \alpha$, and $\sigma_2(w_2, \langle E\rangle_i)$ is defined.

Let $r \subseteq W \times W_2$ be the least relation such that, for all w, w_2, u_2, E, i:

- $r(\tau, \tau_2)$;
- if $R'_i(\tau, \langle E\rangle_i)$ holds then $r(\langle E\rangle_i, \sigma_2(\tau_2, \langle E\rangle_i))$;
- if $r(w, w_2)$ and $S_i(w_2, u_2)$ hold, then $r(\langle\top\rangle_i, u_2)$.

We prove that $M \leqslant_r N$. If $r(w, w_2)$ and $S_i(w_2, u_2)$ hold, then for $u = \langle\top\rangle_i$ we have $r(u, u_2)$ and $R_i(w, u)$. We proceed by showing that if $r(u, u_2)$ and $\alpha \in H(u)$ then $N, \sigma_2, u_2 \vDash \alpha$. The case $u = \tau$ is trivial. Suppose that $u = \langle E\rangle_i$, $r(u, u_2)$, and $\alpha \in H(u)$. There are two cases:

- $u_2 = \sigma_2(\tau_2, \langle E\rangle_i)$ and $R'_i(\tau, \langle E\rangle_i)$; or
- $E = \top$, $r(w, w_2)$, and $S_i(w_2, u_2)$, for some w, w_2.

Consider the first case. Since $\alpha \in H(u)$, either $\Box_i\alpha \in H(\tau)$ or $\langle E\rangle_i\alpha \in H(\tau)$. Hence, $N, \sigma_2, \tau_2 \vDash \Box_i\alpha$ or $N, \sigma_2, \tau_2 \vDash \langle E\rangle_i\alpha$. It follows that $N, \sigma_2, u_2 \vDash \alpha$.

Consider the second case. Since $\alpha \in H(u)$, it follows that $\Box_i\alpha \in H(\tau)$. Hence, $N, \sigma_2, \tau_2 \vDash \Box_i\alpha$. Since $r(w, w_2)$ and $S_i(w_2, u_2)$, it can be shown that u_2 is directly or indirectly reachable from τ_2 (via the accessibility relations S_j, $1 \leqslant j \leqslant m$). Hence $S_i(\tau_2, u_2)$ holds, and $N, \sigma_2, u_2 \vDash \alpha$.

To prove $M \leqslant_r N$, it remains to show that if $r(w, w_2)$ and $R_i(w, u)$ hold, then there exists $u_2 \in W_2$ such that $r(u, u_2)$ and $S_i(w_2, u_2)$ hold. Suppose that $r(w, w_2)$ and $R_i(w, u)$ hold. It follows that $R'_j(\tau, u)$ holds for some $j \leqslant i$. Let $u = \langle E\rangle_j$ and choose $u_2 = \sigma_2(\tau_2, \langle E\rangle_j)$. Thus we have $r(u, u_2)$. Since $r(w, w_2)$, it can be shown that w_2 is directly or indirectly reachable from τ_2 (via the accessibility relations S_k, $1 \leqslant k \leqslant m$). Hence $S_i(w_2, u_2)$ holds. $\Box$

Expected Lemma 5.4. *If $\Box_{i_1} \ldots \Box_{i_h}$ is a $\Box$-lifting form of a modality $\triangle$ in L-normal labeled form and $\triangle$ is an L-instance of $\boxdot$, then $\boxdot\varphi \vDash_L \Box_{i_1} \ldots \Box_{i_h}\varphi$ for any formula φ without labeled modal operators.*

Proof. Just note that $h = 1$ (since $\triangle$ is in L-normal labeled form) and $\Box_{i_1}$ is an L-instance of $\boxdot$. $\Box$

Expected Lemma 5.5. *Let I be an L-normal model generator, M the standard L-model of I, and α a ground L-MProlog goal atom. Suppose that $M \vDash \alpha$. Then α is an L-instance of some atom of $Sat_L(I)$.*

Proof. If α is of the form E or $\Box_i E$, then $\alpha \in Ext_L(I)$ (since $M \vDash \alpha$), and hence $\alpha \in Sat_L(I)$. Suppose that $\alpha = \Diamond_i E$. Let $\langle W, \tau, R_1, \ldots, R_m, H\rangle$ be the standard L-model graph of I. Since $M \vDash \alpha$, there exists a world $u = \langle F\rangle_j$ of M such that $j \leqslant i$ and $E \in H(u)$. By Lemma 5.1, some $\Box$-lifting form of $\langle F\rangle_j E$ belongs to $Ext_L(I)$. It follows that either $\Box_j E$ or $\langle F\rangle_j E$ belongs to $Ext_L(I)$. Hence α is an L-instance of some atom from $Sat_L(I)$. $\Box$

Expected Lemma 5.7. *If P is an L-MProlog program then $P \vDash_L I_{L,P}$.*

Proof. Let $M = \langle D, W, \tau, R_1, \ldots, R_m, \pi\rangle$ be an arbitrary L-model of P and σ a maximal $\Diamond$-realization function on M. Note that if $M, \sigma \vDash \nabla\langle E\rangle_i E$ then $M, \sigma \vDash \langle E\rangle_i E$. It is straightforward to prove by induction on n that $M, \sigma \vDash T_{L,P} \uparrow n$. Hence $M, \sigma \vDash I_{L,P}$. Therefore $P \vDash_L I_{L,P}$. $\Box$

Expected Lemma 5.8. *Let P be an L-MProlog program and I an L-model generator of P. Then the standard L-model of I is an L-model of P.*

Proof. Let M be the standard L-model of I and σ the standard $\Diamond$-realization function on M. It is sufficient to prove that for any ground L-instance $\boxdot(A \leftarrow B_1, \ldots, B_n)$ of some program clause of P, for any $w \in W$ being an L-instance of $\boxdot$, $M, w \vDash (A \leftarrow B_1, \ldots, B_n)$. Suppose that $M, w \vDash B_i$ for all $1 \leqslant i \leqslant n$. We show that $M, w \vDash A$.

Let $\triangle' = w$. We first show that for any ground simple atom B of the form E, $\Box_i E$, or $\Diamond_i E$, if $M, w \vDash B$ then $\triangle' B$ is an L-instance of some atom from $Sat_L(I)$. Suppose that $M, w \vDash B$. If B is of the form E, then by Lemma 5.1, some $\Box$-lifting form of $\triangle' B$ belongs to $Ext_L(I)$, and hence $\triangle' B$ is an L-instance of some atom from $Sat_L(I)$. If B is of the form $\Box_i E$ then, by the construction of M, it follows that $\Box_i E \in Ext_L(I)$, and hence $\{\Box_i E, \Box_m\Box_i E\} \subseteq Sat_L(I)$, which implies that $\triangle' B$ is an L-instance of some atom from $Sat_L(I)$. Now consider the case when B is of the form $\Diamond_i E$. Since $M, w \vDash \Diamond_i E$, either $\Box_j E \in Ext_L(I)$ or $\langle F\rangle_j E \in Ext_L(I)$ for some F and $j \leqslant i$. Hence, either $\{\Box_j E, \Box_m\Box_j E\} \subseteq Sat_L(I)$ or $\{\langle F\rangle_j E, \Box_m\Diamond_j E\} \subseteq Sat_L(I)$ for some F and $j \leqslant i$. Therefore $\triangle' B$ is an L-instance of some atom from $Sat_L(I)$.

Since $M, w \vDash B_i$ for $1 \leqslant i \leqslant n$, it follows that $\triangle' B_i$ is an L-instance of some atom from $Sat_L(I)$. Consequently, $\triangle' A$ is an L-instance of some atom α from $T_{0L,P}(Sat_L(I))$. Suppose that α is in L-normal form. We have $\alpha \in T_{L,P}(I) \subseteq I$. By Lemma 5.2, we have that $M, \sigma \vDash \alpha$, and hence $M, w \vDash A$. Now suppose that α is not in L-normal form, i.e. the length of the modality of α is greater than 1. Thus A is of the form $\Box_i E$ or $\Diamond_i E$. Let A' be the forward labeled form of A. We have $A' \in T_{L,P}(I)$. By Lemma 5.2, it follows that $M, \sigma \vDash A'$. Hence $M, w \vDash A$. $\Box$

Expected Lemma 5.10. *Let $\triangle$ and $\triangle'$ be ground modalities in L-normal labeled form. Let B be an atom of the form E, $\Diamond_i E$, or $\Box_i E$, and B' an atom of the form E, $\Diamond_j E$, $\langle X\rangle_j E$, or $\Box_j E$, where X is a fresh atom variable. Suppose that $\triangle$ is an L-instance of $\triangle'$ and B is an L-instance of B'. Then $\triangle' B'$ is derivable from $\triangle B$ using $rSat_L$.*

Proof. Because $\triangle$ and $\triangle'$ are modalities in L-normal labeled form and $\triangle$ is an L-instance of $\triangle'$, the atom $\triangle' B$ is derivable from $\triangle B$ using the $rSat_L$ rule "$\triangle\nabla_i\alpha \leftarrow \triangle\Box_j\alpha$ if $i \leqslant j$". Next, since B is an L-instance of B', $\triangle' B'$ is derivable from $\triangle' B$ using the first three rules specifying $rSat_L$. $\Box$

Expected Lemma 5.11. *Suppose that β is an atom in almost L-normal labeled form and $\alpha \in Sat_L(\{\beta\})$ or $\alpha \in NF_L(\{\beta\})$. Then there exists an atom β' and a substitution θ s.t. $\beta = \beta'\theta$, the domain of θ consists of fresh atom variables, and β' is derivable from α using $rSat_L$ and rNF_L.*

Proof. We give here a proof only for one representative case, when α is derived from β using the NF_L rule $\nabla\nabla' E \to \nabla' E$, where ∇' is of the form $\Box_i$ or $\langle E\rangle_i$. Suppose that $\alpha = \nabla' E$ and $\beta = \nabla\nabla' E$. If ∇ is of the form $\Box_j$, then by applying the rNF_L rule $\nabla' E \leftarrow \langle X\rangle_j \nabla' E$ and the $rSat_L$ (7) rule instance $\langle X\rangle_j \nabla' E \leftarrow \Box_j \nabla' E$ to α, we obtain $\beta' = \Box_j \nabla' E = \beta$. If ∇ is of the form $\langle F\rangle_j$ (resp. $\langle Y\rangle_j$), then by applying the rNF_L rule $\nabla' E \leftarrow \langle X\rangle_j \nabla' E$ to α, we obtain $\beta' = \langle X\rangle_j \nabla' E$ and have that $\beta = \beta'\theta$, where $\theta = \{X/F\}$ (resp. $\theta = \{X/Y\}$). $\Box$

Expected Lemma 5.12. *Let $\beta = rSat_L(\alpha)$, M be an L-model, σ a $\Diamond$-realization function on M, and θ a substitution. Suppose that $M, \sigma \vDash \forall_c(\beta'\theta)$ for some $\Box$-lifting form β' of β. Then $M, \sigma \vDash \forall_c(\alpha'\theta)$ for some $\Box$-lifting form α' of α.*

Proof. Consider the case when the rule used to derive β from α is $\nabla\nabla' E \leftarrow \nabla' E$, where ∇' is $\Box_i$ or $\Diamond_i$. Let $\alpha = \nabla_j \nabla' E$ and $\beta = \nabla' E$. Then we can choose $\alpha' = \Box_j \nabla' E$. It is easily seen that $M, \sigma \vDash \forall_c(\alpha'\theta)$, since $M, \sigma \vDash \forall_c(\beta'\theta)$. Now consider the case when the rule used to derive β from α is $\triangle\Diamond_i E \leftarrow \triangle\Diamond_j E$ with $i > j$. Let $\alpha = \triangle\Diamond_i E$, $\beta = \triangle\Diamond_j E$, and $\beta' = \triangle'\nabla_j E$. Then we can choose $\alpha' = \triangle'\Diamond_i E$. Since $M, \sigma \vDash \forall_c(\beta'\theta)$, we have $M, \sigma \vDash \forall_c(\alpha'\theta)$. The two remaining cases are similar to the last case.

Expected Lemma 5.13. *Let $\beta =_\delta rNF_L(\alpha)$, M be an L-model, σ a maximal $\Diamond$-realization function on M, and θ a substitution. Suppose that $M, \sigma \vDash \forall_c(\beta'\theta)$ for some $\Box$-lifting form β' of β. Then $M, \sigma \vDash \forall_c(\alpha'\delta\theta)$ for some $\Box$-lifting form α' of α.*

Proof. There is only one rNF_L rule. Let $\alpha\delta = \nabla E$ and $\beta = \langle X\rangle_j \nabla E$, where ∇ is $\Box_i$ or $\langle E\rangle_i$. If $\nabla = \Box_i$, then let $\nabla' = \Box_i$, else let $\nabla' = \Diamond_i$. Since $M, \sigma \vDash \forall_c(\beta'\theta)$, we have $M \vDash \nabla' E\theta$. Since σ is a maximal $\Diamond$-realization function on M, it follows that $M, \sigma \vDash \forall_c(\nabla E\theta)$. Hence we can choose $\alpha' = \alpha$. $\Box$

8. Programming in MProlog for multiagent systems

To program for multiagent systems we can use the logics $KD4_s5_s$, $KD45_{(m)}$, and $KD4I_g5_a$. In these logics, $\Box_i\varphi$ stands for "agent i believes that φ is true", while $\Diamond_i\varphi$ stands for "φ is considered possible by agent i". The logic $KD4_s5_s$ can be used for distributed systems of belief, in which agents have full access to belief bases of each other. The logics $KD45_{(m)}$ and $KD4I_g5_a$ are intended for reasoning about epistemic states of agents. In $KD4I_g5_a$, some modal indices stand for groups of agents, and using them we can reason about common belief. In this section, we present a schema for semantics of $KD4_s5_s$-MProlog. Schemata for semantics of MProlog in $KD45_{(m)}$ and $KD4I_g5_a$ are presented in the Appendix, and proofs of their correctness are given in [32].

8.1. A schema for semantics of $KD4_s5_s$-MProlog

In this subsection L denotes $KDI4_s5$. It can be checked that a connected frame $\langle W, \tau, R_1, \ldots, R_m\rangle$ is a $KD4_s5_s$-frame iff there are nonempty subsets of worlds $W_1, \ldots, W_m$ such that $W = \{\tau\} \cup W_1 \cup \cdots \cup W_m$ and $R_i = W \times W_i$, for

$1 \leqslant i \leqslant m$. Note that this property is similar to the property of $KDI4_s5$-frames. The difference is that the logic $KD4_s5_s$ does not contain the axiom (I) and in this logic we do not have the condition that $W_i \subseteq W_j$ for $i < j$.

In Table 3, we present a schema for semantics of $KD4_s5_s$-MProlog. The L-normal form of modalities and the rules (1)–(4) and (7) in that schema are justified by the L-tautology $\nabla\varphi \equiv \nabla'\nabla\varphi$ with ∇ and ∇' being unlabeled modal operators, while the rule (6) is based on the axiom (D). This schema is similar to the schema for semantics of $KDI4_s5$-MProlog, except that it does not contain rules involving with the axiom (I). Analogously as for $KDI4_s5$, we can prove the following theorem.

Theorem 8.1. *The schema given in Table 3 for semantics of $KD4_s5_s$-MProlog is correct.*

Example 8.1. Reconsider the MProlog program P_{ddb} given in Example 4.2. To increase readability, we recall some clauses of P_{ddb}:

$$\varphi_1 = \Box_1 likes(Jan, cola) \leftarrow$$
$$\varphi_5 = \Box_2 likes(Jan, pepsi) \leftarrow$$
$$\varphi_8 = \Box_2(likes(x, cola) \leftarrow likes(x, pepsi))$$
$$\varphi_{10} = \Box_3 likes(Jan, cola) \leftarrow$$
$$\varphi_{13} = \Box_3(very\_much\_likes(x, y) \leftarrow likes(x, y), \Box_1 likes(x, y), \Box_2 likes(x, y))$$
$$\varphi_{14} = very\_much\_likes(x, y) \leftarrow \Box_3 very\_much\_likes(x, y).$$

Here is an SLD-refutation of $P \cup \{\leftarrow very\_much\_likes(x, y)\}$ in $KD4_s5_s$:

Goals	Input clauses/rules	mgu's
$\leftarrow very\_much\_likes(x, y)$		
$\leftarrow \Box_3 very\_much\_likes(x, y)$	φ_{14}	$\{x_1/x, y_1/y\}$
$\leftarrow \Box_3 likes(x, y), \Box_3\Box_1 likes(x, y), \Box_3\Box_2 likes(x, y)$	φ_{13}	$\{x_2/x, y_2/y\}$
$\leftarrow \Box_3\Box_1 likes(Jan, cola), \Box_3\Box_2 likes(Jan, cola)$	φ_{10}	$\{x/Jan, y/cola\}$
$\leftarrow \Box_1 likes(Jan, cola), \Box_3\Box_2 likes(Jan, cola)$	(7)	ε
$\leftarrow \Box_3\Box_2 likes(Jan, cola)$	φ_1	ε
$\leftarrow \Box_2 likes(Jan, cola)$	(7)	ε
$\leftarrow \Box_2 likes(Jan, pepsi)$	φ_8	$\{x_7/Jan\}$
$\Diamond$	φ_5	ε.

The schema given in Table 3 is formulated so that it can use the proofs given in Section 5. However, similarly as for the case of $KDI4_s5$, the rules (5) and (6) of Table 3 can be simplified in the way that the occurrences of $\triangle$ in those rules are deleted and α in the rule (6) is replaced by E. Furthermore, when resolving a goal with an input clause, if we relax the condition that the mgu θ unifies the selected head atom A' with the forward labeled form A'' of the head of the input clause, but only require that θ is a most general substitution such that $A'\theta$ and $A''\theta$ have the same classical atom and $A'\theta$ is an L-instance of $A''\theta$, then the rule (6) can be deleted. It can be shown that every SLD-refutation in the original calculus can be simulated in the new calculus by another one with the same computed answer. This means that the new SLD-resolution calculus is also sound and complete.

An agent should keep clauses that define its epistemic states. This means that agent i should keep clauses of the form $\nabla_i E \leftarrow B_1, \ldots, B_n$ or $\Box_i(A \leftarrow B_1, \ldots, B_n)$. Furthermore, program clauses of the form $\Box_i(\Box_j E \leftarrow B_1, \ldots, B_n)$ with $i \neq j$ have little sense in distributed systems of belief. It can be shown that program clauses of that form can be disallowed without reducing expressiveness of $KD4_s5_s$-MProlog. If we adopt this restriction then the rule (4) in Table 3 can be modified so that the involved modal operators have the same modal index (i.e. agent index). Program clauses of the form $E \leftarrow B_1, \ldots, B_n$ can be kept by a special agent, which communicates with users. Whenever an agent meets a goal atom of the form $\nabla_i E$ it will require agent i to solve the goal $\leftarrow \nabla_i E$, and whenever an agent meets a goal atom of the form E (without modal context) it will require the special agent to solve the goal $\leftarrow E$.

Table 3
A schema for semantics of $KD4_s5_s$-MProlog

$L = KD4_s5_s$, L-MProlog		
$\preccurlyeq_L$ is defined by Definition 5.12.		
A modality is in L-normal form if its length $\leqslant 1$.		
Rules specifying		
Ext_L	No rules	
Sat_L	$\Box_i E \to \Box_j \Box_i E$	(1)
	$\langle F \rangle_i E \to \Box_j \Diamond_i E$	(2)
NF_L	$\nabla\nabla' E \to \nabla' E$ if ∇' is of the form $\Box_i$ or $\langle E \rangle_i$	(3)
rNF_L	$\nabla E \leftarrow \langle X \rangle_j \nabla E$ if ∇ is of the form $\Box_i$ or $\langle E \rangle_i$ and X is a fresh atom variable	(4)
$rSat_L$	$\triangle \Diamond_i E \leftarrow \triangle \langle X \rangle_i E$ for X being a fresh atom variable	(5)
	$\triangle \nabla_i \alpha \leftarrow \triangle \Box_i \alpha$	(6)
	$\nabla\nabla' E \leftarrow \nabla' E$ if ∇' is of the form $\Box_i$ or $\Diamond_i$	(7)

9. Discussion and conclusion

9.1. Relation with other works

Our framework is formulated with an intention for multimodal logics whose frame restrictions consist of the conditions of seriality and some classical first-order Horn clauses. In particular, we have applied the framework for the *BSMM* class of basic serial multimodal logics. Clarity of the $Sat_L/rSat_L$ rules used for the given schema for semantics of *BSMM*-MProlog suggests that our framework can be applied for other multimodal logics not belonging to the *BSMM* class. For example, it can be instantiated for serial context-free grammar logics, which are multimodal logics characterized by the axioms of seriality and axioms of the form $\Box_i \varphi \to \Box_{j_1} \ldots \Box_{j_k} \varphi$.

In multimodal logic programming, Debart et al. [15] considered multimodal logics which have a finite number of modal operators $\Box_i$ and $\Diamond_i$ of any type among *KD*, *KT*, *KD*4, *KT*4, *KF* and interaction axioms of the form $\Box_i \varphi \to \Box_j \varphi$. This class is relatively smaller than the *BSMM* class considered in this work. Namely, apart from the axiom $(F) : \Box_i \varphi \equiv \Diamond_i \varphi$, the other modal axioms considered by Debart et al. in [15] are included for the *BSMM* class, while the symmetry modal axioms (B) and (5) and interaction axioms other than (I) like $\Box_i \varphi \to \Box_j \Box_k \varphi$ are absent in the work by Debart et al. [15]. In our opinion, the approach by Debart et al. can be generalized to deal with the *BSMM* class. However, it is not clear to us whether such an extension is straightforward or not: for example, are there only finitely many (maximally general) unifiers for any two "paths" in any *BSMM* logic?

Another work explicitly devoted to multimodal logic programming is by Baldoni et al. [10]. The authors gave a framework for developing declarative and operational semantics for logic programs in multimodal logics which have axioms of the form $[t_1] \ldots [t_n]\varphi \to [s_1] \ldots [s_m]\varphi$, where $[t_i]$ and $[s_j]$ are universal modal operators indexed by terms t_i and s_j, respectively. To represent worlds in canonical models of programs, the authors used sequences of universal modal operators, which are similar to sequences of $\langle \top \rangle_i$ in our work. The work [10] contains several interesting examples (illustrating "epistemic reasoning, defining parametric and nested modules, describing inheritance in a hierarchy of classes and reasoning about actions"). The logics considered in [10] are called inclusion multimodal logics (also known as grammar logics). This class of logics is disjoint with the class of multimodal logics considered in this work. Namely, the former multimodal logics are not serial, while the latter ones are serial. However, the biggest difference between [10] and our work is that these two works base on different settings. Baldoni et al. [10] assume that modal logic programs and goals do not contain existential modal operators, while we do not adopt such a restriction. Our framework cannot cope with context-sensitive grammar logics, while the framework by Baldoni et al. [10] does not consider reasoning about possibility. [12]

Despite that Nonnengart [38] studied modal logic programming explicitly only for serial monomodal logics, his semi-functional translation method works also for serial multimodal logics. As mentioned earlier, Nonnengart [38] uses accessibility relations for translated programs, but with optimized clauses for representing properties of the accessibility relations, and does not modify unification.

In our opinion, all the mentioned approaches are worth for studying. Each approach offers a method to deal with modalities, which in turn can be exploited deeply or not. For example, using semi-functional translation, one can use the restrictions on accessibility relations without optimizations. But in that case, the proof procedure would not be very efficient. As another example, although the logic $KDI4_s5$ belongs to the *BSMM* class, our SLD-resolution calculus given for $KDI4_s5$-MProlog is much more efficient than our SLD-resolution calculus given for *BSMM*-MProlog when used for $KDI4_s5$.

The direct approach has a good property that it is somehow friendlier for users than the translation approaches in the debugging and iterative modes of programming. Let us consider, for example, translation of the goals $G_1 = \leftarrow \Box p$ and $G_2 = \leftarrow \Box\Diamond p(x)$. Using any of the mentioned translation methods, G_1 is translated to $\leftarrow p(\tau : a)$. The goal G_2 is translated to $\leftarrow p(\tau : f(x) : y, x)$ using the functional translation, and to $\leftarrow p(y, x), R(\tau : f(x), y)$ using the semi-functional translation. In our opinion, the translated goals are much less intuitive than the original ones. With a similar opinion, a reviewer of our conference paper [34] wrote "it is important not to translate away all modalities because the modalities allow us to separate object-level and epistemic-level notions nicely". Furthermore, if we want to let programmers to have some control in using properties of the base logic, then rules used in our approach (e.g. in the form $\triangle\Box_j\Diamond_k\alpha \leftarrow \triangle\Diamond_i\alpha$ or $\triangle\Box_i\alpha \leftarrow \triangle\Diamond_j\Box_k\alpha$) are more intuitive for them than rules used in the semi-functional translation approach (e.g. in the form $R_k(x, y) \leftarrow R_j(z, x), R_i(z, y)$).

Note that our approach and the translation approaches all assume the conditions of seriality. With respect to the least model semantics, the semi-functional translation has the good property that it is straightforward to convert the least Herbrand model of a translated program to the least Kripke model of the original program. It seems hard to develop the least Kripke model semantics for modal logic programs using the functional translation approach. With respect to fixed/varying domain and rigid/flexible terms, Debart et al. [15] used Kripke semantics with fixed domain and rigid/flexible terms. Nonnengart [38] used Kripke semantics with varying domain and flexible terms. Baldoni et al. [10] used Kripke semantics with varying domain and rigid terms. In this work, we used Kripke semantics with fixed domain and rigid terms. See Garson's work [22] for a survey of the different systems for quantified modal logic. A discussion on extending our framework for the other versions of Kripke semantics is given later.

In comparison with other works that also use the direct approach for defining declarative and procedural semantics for modal logic programs, e.g. [6,10], our work [31] and this are the first ones that do not assume any special restriction on occurrences of modal operators. In [6] Balbiani et al. gave a declarative semantics and an SLD-resolution for a class of logic programs in the monomodal logics *KD*, *T* and *S*4. To modal programs the authors associate a declarative semantics represented by a tree which is defined as the limit of a certain transformation on modal programs. The fixpoint represents a minimal Kripke model of the program. The work assumes that the $\Box$ operator does not occur in bodies of program clauses and goals. In the definition of the minimal Kripke model of a program [6], the technique of connecting each newly created world to an empty world at the time of its creation (or a similar one) is not used, hence although the minimal Kripke model of a program defined in [6] is minimal with respect to the restricted class of goals, in general it is not a least Kripke model of the program in the considered logic. There is a common point between [6] and our work: in both of the works, labeled modal operators are used to convert $\langle t\rangle(\varphi \wedge \psi)$ to $\langle t\rangle\varphi \wedge \langle t\rangle\psi$. Labeled modal operators in [6] come from Skolemization, and terms are used to label the $\Diamond$ operator. In our work, the labeling

[12] Note that every positive propositional logic program without $\Diamond$ in $KD45$ (i.e. $KD45_{(m)}$ with $m = 1$) has a least $KD45$-model with *two* possible worlds, and it cannot express complicated properties about possibility. Furthermore, existential modal operators cannot be totally replaced by universal modal operators using interaction axioms. For example, every positive propositional logic program without existential modal operators has a least $KDI4_s5$-model with $m + 1$ possible worlds (recall that m is the number of different modal indices), and we have the same problem as stated before.

technique results from the technique of building model graphs, and we feel convenient to use classical atoms and atom variables to label $\Diamond_i$ operators.

In comparison with our previous work [31] on monomodal logic programming, in this work the operators Ext_L, Sat_L, NF_L, $rSat_L$, and rNF_L are all specified by sets of rules. This way is more declarative and better reflects axioms of the base logic. The $\Box$-lifting and backward labeling operators introduced in [31] are classified in this work as rules for specifying $rSat_L$. The definitions of L-instance of an atom and L-instance of a program clause have been also abstracted. The framework given here differs from [31] at an important aspect that it is formulated for a class of modal logics but not for specific modal logics. At least, the proofs of soundness and completeness of SLD-resolution given in Section 5.5 are reusable without modifications. The framework can be easily instantiated for the serial monomodal logics considered in [31].

In the technical report [32], we study also the case when existential modal operators are disallowed in MProlog programs and goals, resulting in MProlog-$\Box$, and show that in that case schemata for semantics of MProlog can be significantly simplified.

This work extends or relates to our recent conference papers [33–37].

9.2. On implementation of MProlog

As far as we know, amongst the works by other authors that use the direct approach for modal logic programming, only the Molog system proposed by Fariñas del Cerro [18] has been implemented. With Molog, the user can fix a modal logic and define or choose the rules to deal with modal operators. Molog can be viewed as a framework which can be instantiated with particular modal logics. As an extension of Molog, the *Toulouse Inference Machine* (TIM) [7] (together with an abstract machine model called TARSKI for implementation [8]) makes it possible for a user to select clauses which cannot exactly unify with the current goal, but just resemble it in some way.

As reported in [33,34], we have designed and implemented the modal logic programming system MProlog using our framework. This system is written in Prolog as a module for Prolog. Codes, libraries, and most features of Prolog can be used in MProlog programs. The system contains a number of built-in SLD-resolution calculi for different modal logics, including all of the considered multimodal logics of belief and basic serial monomodal logics. It has been designed so that users can implement and add SLD-resolution calculi to the system in a modular way.

Users can use and mix different calculi in an MProlog program. For flexibility, there are three kinds of predicates: modal predicates, classical predicates (which do not depend on possible worlds in Kripke models), and classical predicates that are defined using also modal predicates. The last kind of predicates is useful, for example, when a predicate is implemented by different programmers for different modules, and each module uses a different modal logic.

Technically, modalities are represented as lists. For example, $\Box_i \langle X \rangle_3 \Diamond_j p(a)$ may be represented as $[bel(I), pos(3, X), pos(J)] : p(a)$, where *bel* stands for “believes”, and *pos* for “possible”. Notations of modal operators depend on how the base SLD-resolution calculus is defined. As another example, for MProlog-$\Box$ (which disallows existential modal operators in programs and goals), we can represent $\Box_{i_1} \ldots \Box_{i_k}$ as $[I_1, \ldots, I_k]$.

Backward rules can be of the form “*AtomIn :- PreCondition, AtomOut, PostComputation.*” with *AtomIn* and *AtomOut* being atoms of the form $M : E$, where M (standing for a modality) and E (standing for a classical atom) may be variables in Prolog, and M may also be a list; *PreCondition* and *PostComputation* are (possibly empty) sequences of formulas in Prolog separated by “,”.

For the solver of MProlog, a resolving cycle is defined to be a derivation using a sequence of $rSat_L/rNF_L$ rules and a program clause. Shorter sequences of rules are tried before longer ones. Programmers have access to the history of the current resolving cycle.

For effectiveness, classical fragments in MProlog programs are interpreted by Prolog itself, and there are a number of features that can be used to restrict the search space.

The implemented MProlog system has a very different theoretical foundation than Molog. In MProlog, the labeling technique is used for existential modal operators instead of Skolemization. Our system uses new technicalities like normal forms of modalities and pre-orders between modal operators. MProlog also eliminates some drawbacks of Molog, e.g. MProlog gives computed answers, while Molog can only answer “yes” or “no”.

For further details on the implemented MProlog system, we refer the reader to [34].

9.3. Concluding remarks

We used *fixed-domain* Kripke models with *rigid terms* for the framework. This is the most common choice, but can we loose the restrictions of fixed domain and rigid terms? Since we do not use equalities in MProlog programs, the restriction of rigid terms is not essential. What happens if we allow varying domains? First, we define a *varying-domain Kripke model* to be a tuple $M = \langle D, W, \tau, R_1, \ldots, R_m, \pi\rangle$, where for each $w \in W$, $D(w)$ is a set called the *domain of* w, $\langle W, \tau, R_1, \ldots, R_m\rangle$ is a Kripke frame, and for each $w \in W$, $\pi(w)$ is an interpretation of constant symbols, function symbols and predicate symbols on the domain $D(w)$. Second, a *variable assignment* V w.r.t. M is a function that maps each pair of a world w and a variable x to an element of the domain of w. The value of $t^{M,w}[V]$ for a term t at a world w of M is defined as usual. According to these definitions, terms are flexible. The satisfaction relation is then defined in the usual way, except that

$M, V, w \vDash p(t_1, \ldots, t_n)$ iff $(t_1^{M,w}[V], \ldots, t_n^{M,w}[V]) \in \pi(w)(p)$;
$M, V, w \vDash \forall x.\varphi$ iff for all V' different from V only for pairs $(\_, x)$, $M, V', w \vDash \varphi$
$M, V, w \vDash \exists x.\varphi$ iff there exists V' different from V only for pairs $(\_, x)$ s.t. $M, V', w \vDash \varphi$.

Our thesis is that the framework can be easily adapted for varying-domain Kripke models. Informal argumentations for this are: first, we do not use the Barcan formula $\forall x.\Box_i \varphi \to \Box_i \forall x.\varphi$ and the converse Barcan formula $\Box_i \forall x.\varphi \to \forall x.\Box_i \varphi$ in any way. Second, as we consider only positive modal logic programs without equality, the method of constructing least Kripke models for positive modal logic programs still works for the case of varying-domain Kripke models. Precise analysis, however, should be done for this problem.

In [36], basing on the fixpoint semantics presented in this work, we developed modal relational algebras and advanced computational methods like the magic-set transformation for modal deductive databases. When dealing with modal deductive databases, the direct approach has an advantage over the translation approaches. Given an MDatalog program, which is an MProlog program without function symbols and consisting of *allowed* [13] program clauses, the translation methods translate it to a program that may contain Skolem function symbols and *disallowed* program clauses, which is undesirable.

One of the good features of our framework is L-normal form of modalities. In logics like $KDI4_s5$, $KDI45$, $KD4_s5_s$, $KD45_{(m)}$, it is a tool allowing us to restrict lengths of modalities appearing in derivations. Such a tool was not introduced in [6,1,15,38,10]. Due to L-normal form of modalities, in [36] we were able to show that the intentional relations of a modal deductive database in $L \in \{KDI4_s5, KDI45, KD4_s5_s, KD45_{(m)}\}$ can be computed in PTIME and have polynomial size (in the size of the extensional relations).

When dealing with modal logic programs with negation, the translation approaches give rise to the floundering problem [14] even when the input modal logic program and goal are *allowed*. [15] To see this, just consider the program clause $p \leftarrow \Diamond\neg q$. Extending our direct approach for dealing with negation is also a hard problem. However, we think that it is possible to overcome the difficulty and we will study this problem in the near future.

Our most important contribution in this work is the framework for developing fixpoint semantics, the least model semantics, and SLD-resolution calculi for multimodal logic programs. The framework is formulated in a direct way (not using translation to the classical logic) and closely to the style of classical logic programming. It is applicable and useful for a wide class of modal logics, including *BSMM* logics, serial context-free grammar logics, and the basic serial monomodal logics. The framework allows not only to exploit syntactic properties of the base logic, as in the case of *BSMM*, but also to use semantical properties of the base logic, as in the case of $KDI4_s5$.

In literature of computer science, multimodal logics are much more studied for reasoning about knowledge than about belief (see, e.g. [17,28]). In this work, we have concentrated on multimodal logics intended for reasoning about belief, in particular, for reasoning about multidegree belief, for use in distributed systems of belief, and for reasoning about epistemic states of agents in multiagent systems. The logics of multidegree belief proposed by us are somehow similar to graded modal logics but different at the aspect that degrees in the former case are symbolic, while grades

[13] A program clause is *allowed* if all of its variables occur (also) in the body.
[14] Which occurs when a derived goal contains only nonground negative literals.
[15] In the sense that every variable occurring in a clause occurs also in a positive literal of the body of the clause.

in the latter case are numeric.[16] We think that our schemata for semantics of MProlog in the considered multimodal logics of belief are practically useful. On the other hand, our schema for semantics of *BSMM*-MProlog is interesting from the theoretical point of view. It shows that declarative and procedural semantics of multimodal logic programs can be formulated in a direct way, not using translation to the classical logic. These schemata are another one of our main contributions.

In summary, we have successfully applied the direct approach for modal logic programming in a large class of multimodal logics, while not assuming any special restriction on the form of logic programs and goals.

Acknowledgements

I would like to thank professor Andrzej Szałas for reading the draft of this paper and giving helpful comments. My thanks go also to the anonymous reviewers for helpful comments and suggestions.

Appendix A. Some schemata for semantics of *L*-MProlog

Some schemata for semantics of MProlog are given in Tables 4–7.

Table 4
A schema for semantics of *KDI*45-MProlog

$L = KDI45$, L-MProlog

$\preccurlyeq_L$ is defined by Definition 5.12 in Section 5.3.

$\nabla^{(1)}_{i_1} \ldots \nabla^{(k)}_{i_k}$ is in L-normal form if $i_1 > \ldots > i_k$

The following rules are accompanied with the condition that the atoms in both sides are in L-normal labeled form for the rules specifying Ext_L and in almost L-normal labeled form for the other rules. (*)

Ext_L	$\triangle\Box_i\alpha \to \triangle\Box_j\alpha$ if $i > j$	(1)
	$\triangle\Box_i\alpha \to \triangle\Box_i\Box_j\alpha$ if $i > j$	(2)
	$\triangle\Box_i\Box_j\alpha \to \triangle\Box_j\alpha$ if $i > j$	(3)
Sat_L	The rules for Ext_L with the modification stated in (*), plus	
	$\triangle\Box_i E \to \triangle\Box_i\Box_i E$	(4)
	$\triangle\nabla E \to \triangle\Box_i\Diamond_i E$ if $\Diamond_i \preccurlyeq_L \nabla$	(5)
	$\triangle\Box_i\nabla_j E \to \triangle\Diamond_j E$ if $i > j$	(6)
	$\triangle\langle F\rangle_i\nabla_j E \to \triangle\Diamond_i E$ if $i > j$	(7)
NF_L	$\triangle\nabla_i\nabla'_j E \to \triangle\nabla'_j E$ if ∇'_j is of the form $\Box_j$ or $\langle E\rangle_j$ and $i \leqslant j$	(8)
rNF_L	$\triangle\nabla_j E \leftarrow \triangle\langle X\rangle_i\nabla_j E$ if ∇_j is of the form $\Box_j$ or $\langle E\rangle_j$, X is a fresh atom variable, and $i \leqslant j$	(9)
$rSat_L$	$\triangle\Diamond_i E \leftarrow \triangle\langle X\rangle_i E$ for X being a fresh atom variable	(10)
	$\triangle\nabla_i\alpha \leftarrow \triangle\Box_j\alpha$ if $i \leqslant j$	(11)
	$\triangle\Diamond_i E \leftarrow \triangle\Diamond_j E$ if $i > j$	(12)
	$\triangle\Box_i\Box_j\alpha \leftarrow \triangle\Box_i\alpha$ if $i \geqslant j$	(13)
	$\triangle\Box_i\Diamond_i E \leftarrow \triangle\Diamond_i E$	(14)
	$\triangle\Box_i\alpha \leftarrow \triangle\Box_j\Box_i\alpha$ if $i < j$	(15)
	$\triangle\Diamond_i E \leftarrow \triangle\langle X\rangle_i\Diamond_i E$ for X being a fresh atom variable	(16)

[16] Grades are used to indicate the number of worlds accessible from the current world.

Table 5
Schemata for semantics of MProlog in $KDI4_s$ and $KDI4$

$L = KDI4_s$, L-MProlog		
$\preccurlyeq_L$ is defined by Definition 5.12 in Section 5.3.		
No restrictions on L-normal form of modalities No rules specifying NF_L and rNF_L		
Rules specifying		
Ext_L	$\triangle\Box_i\alpha \to \triangle\Box_j\alpha$ if $i > j$	(1)
	$\triangle\Box_i\alpha \to \triangle\Box_j\Box_i\alpha$	(2)
Sat_L	The rules specifying Ext_L plus	
	$\triangle\nabla\nabla' E \to \triangle\Diamond_i E$ if $\Diamond_i \preccurlyeq_L \nabla'$	(3)
$rSat_L$	$\triangle\Diamond_i E \leftarrow \triangle\langle X\rangle_i E$ for X being a fresh atom variable	(4)
	$\triangle\nabla_i\alpha \leftarrow \triangle\Box_j\alpha$ if $i \leqslant j$	(5)
	$\triangle\Diamond_i E \leftarrow \triangle\Diamond_j E$ if $i > j$	(6)
	$\triangle\nabla\Box_i\alpha \leftarrow \triangle\Box_i\alpha$	(7)
	$\triangle\Diamond_i E \leftarrow \triangle\langle X\rangle_j\Diamond_i E$ for X being a fresh atom variable	(8)
$L = KDI4$, L-MProlog		
$\preccurlyeq_L$ is defined by Definition 5.12 in Section 5.3.		
No restrictions on L-normal form of modalities No rules specifying NF_L and rNF_L		
Rules specifying		
Ext_L	$\triangle\Box_i\alpha \to \triangle\Box_j\alpha$ if $i > j$	(1)
	$\triangle\Box_i\alpha \to \triangle\Box_i\Box_i\alpha$	(2)
Sat_L	The rules specifying Ext_L plus	
	$\triangle\nabla\nabla' E \to \triangle\Diamond_i E$ if $\Diamond_i \preccurlyeq_L \nabla$ and $\Diamond_i \preccurlyeq_L \nabla'$	(3)
$rSat_L$	$\triangle\Diamond_i E \leftarrow \triangle\langle X\rangle_i E$ for X being a fresh atom variable	(4)
	$\triangle\nabla_i\alpha \leftarrow \triangle\Box_j\alpha$ if $i \leqslant j$	(5)
	$\triangle\Diamond_i E \leftarrow \triangle\Diamond_j E$ if $i > j$	(6)
	$\triangle\Box_i\Box_i\alpha \leftarrow \triangle\Box_i\alpha$	(7)
	$\triangle\Diamond_i E \leftarrow \triangle\langle X\rangle_j\Diamond_i E$ for $i \geqslant j$ and X being a fresh atom variable	(8)

Table 6
A schema for semantics of $KD45_{(m)}$-MProlog

$L = KD45_{(m)}$, L-MProlog

$\preccurlyeq_L$ is defined by Definition 5.12 in Section 5.3

$\nabla^{(1)}_{i_1} \ldots \nabla^{(k)}_{i_k}$ is in L-normal form if $i_j \neq i_{j+1}$ for all $1 \leqslant j < k$

Table 6 (*contd.*)

Both sides of each rule given below are in almost L-normal labeled form		
Ext_L	No rules	
Sat_L	$\triangle\Box_i E \to \triangle\Box_i\Box_i E$	(1)
	$\triangle\langle F\rangle_i E \to \triangle\Box_i\Diamond_i E$	(2)
NF_L	$\triangle\nabla_i\nabla'_i E \to \triangle\nabla'_i E$ if ∇'_i is of the form $\Box_i$ or $\langle E\rangle_i$	(3)
rNF_L	$\triangle\nabla_i E \leftarrow \triangle\langle X\rangle_i\nabla_i E$ if ∇'_i is of the form $\Box_i$ or $\langle E\rangle_i$ and X is a fresh atom variable	(4)
$rSat_L$	$\triangle\Diamond_i E \leftarrow \triangle\langle X\rangle_i E$ for X being a fresh atom variable	(5)
	$\triangle\nabla_i\alpha \leftarrow \triangle\Box_i\alpha$	(6)
	$\triangle\nabla_i\nabla'_i E \leftarrow \triangle\nabla'_i E$ if ∇'_i is of the form $\Box_i$ or $\Diamond_i$	(7)

Table 7
A schema for semantics of $KD4I_g5_a$-MProlog

$L = KD4I_g5_a$, L-MProlog		
$\preccurlyeq_L$ is defined by Definition 5.12 in Section 5.3.		
$\nabla^{(1)}_{i_1}\ldots\nabla^{(k)}_{i_k}$ is in L-normal form if for all $1 \leqslant j < k$ if $g(i_j)$ is a singleton then $i_j \neq i_{j+1}$		
Both sides of each rule given below are in almost L-normal labeled form		
Ext_L	$\triangle\Box_i\alpha \to \triangle\Box_j\alpha$ if $g(i) \supset g(j)$	(1)
	$\triangle\Box_i\alpha \to \triangle\Box_i\Box_i\alpha$	(2)
Sat_L	The rules specifying Ext_L plus	
	$\triangle\langle F\rangle_i E \to \triangle\Box_i\Diamond_i E$ if $g(i)$ is a singleton	(3)
	$\triangle\nabla\nabla' E \to \triangle\Diamond_i E$ if $\Diamond_i \preccurlyeq_L \nabla$ and $\Diamond_i \preccurlyeq_L \nabla'$	(4)
NF_L	$\triangle\nabla_i\nabla'_i E \to \triangle\nabla'_i E$ if $g(i)$ is a singleton and ∇'_i is of the form $\Box_i$ or $\langle E\rangle_i$	(5)
rNF_L	$\triangle\nabla_i E \leftarrow \triangle\langle X\rangle_i\nabla_i E$ if $g(i)$ is a singleton, ∇_i is of the form $\Box_i$ or $\langle E\rangle_i$, and X is a fresh atom variable	(6)
$rSat_L$	$\triangle\Diamond_i E \leftarrow \triangle\langle X\rangle_i E$ for X being a fresh atom variable	(7)
	$\triangle\nabla_i\alpha \leftarrow \triangle\Box_j\alpha$ if $g(i) \subseteq g(j)$	(8)
	$\triangle\Diamond_i E \leftarrow \triangle\Diamond_j E$ if $g(i) \supset g(j)$	(9)
	$\triangle\Box_i\Box_i\alpha \leftarrow \triangle\Box_i\alpha$	(10)
	$\triangle\nabla_i\Diamond_i E \leftarrow \triangle\Diamond_i E$ if $g(i)$ is a singleton	(11)
	$\triangle\Diamond_i E \leftarrow \triangle\langle X\rangle_j\Diamond_i E$ if $g(i) \supseteq g(j)$ and X is a fresh atom variable	(12)

References

[1] S. Akama, A meta-logical foundation of modal logic programming, Japan, December 1989.

[2] H. Aldewereld, W. van der Hoek, J.-J.Ch. Meyer, Rational teams: logical aspects of multi-agent systems, Fund. Inform. 63 (2–3) (2004) 159–183.

[3] K.R. Apt, Logic programming, in: J. van Leeuwen (Ed.), Handbook of Theoretical Computer Science, Volume B: Formal Models and Semantics, Elsevier, Amsterdam, 1990.

[4] K.R. Apt, M.H. van Emden, Contributions to the theory of logic programming, J. ACM 29 (3) (1982) 841–862.

[5] G. Attardi, M. Simi, Proofs in context, in: J. Doyle, E. Sandewall, P. Torasso (Eds.), KR'94: Principles of Knowledge Representation and Reasoning, Morgan Kaufmann, San Francisco, 1994, pp. 16–26.
[6] Ph. Balbiani, L. Fariñas del Cerro, A. Herzig, Declarative semantics for modal logic programs, in: Proc. 1988 Internat. Conf. on Fifth Generation Computer Systems, ICOT, 1988, pp. 507–514.
[7] Ph. Balbiani, A. Herzig, M. Lima-Marques, TIM: the Toulouse inference machine for non-classical logic programming, in: PDK'91: International Workshop on Processing Declarative Knowledge, Springer, Berlin, 1991, pp. 365–382.
[8] Ph. Balbiani, A. Herzig, M. Lima-Marques, Implementing Prolog extensions: a parallel inference machine, in: Proc. 1992 Internat. Conf. on Fifth Generation Computer Systems, ICOT, 1992, pp. 833–842.
[9] M. Baldoni, Normal multimodal logics with interaction axioms, in: D. Basin, M. D'Agostino, D.M. Gabbay, L. Viganò (Eds.), Labelled Deduction, Kluwer Academic Publishers, Dordrecht, MA, 2000, pp. 33–57.
[10] M. Baldoni, L. Giordano, A. Martelli, A framework for a modal logic programming, in: Joint Internat. Conf. Symp. on Logic Programming, MIT Press, Cambridge, MA, 1996, pp. 52–66.
[11] M. Baldoni, L. Giordano, A. Martelli, Translating a modal language with embedded implication into Horn clause logic, in: R. Dyckhoff, H. Herre, P. Schroeder-Heister (Eds.), Proc. of ELP'96, Lecture Notes in Computer Science, Vol. 1050, 1996, pp. 19–33.
[12] P. Blackburn, M. de Rijke, Y. Venema, Modal Logic, Cambridge University Press, Cambridge, MA, 2002.
[13] A. Cimatti, L. Serafini, Multi-agent reasoning with belief contexts: the approach and a case study, in: M. Wooldridge, N.R. Jennings (Eds.), Proc. of ECAI-94, Lecture Notes in Computer Science, Vol. 890, Springer, Berlin, 1995, pp. 71–85.
[14] M.J. Cresswell, G.E. Hughes, A New Introduction to Modal Logic, Routledge, London, 1996.
[15] F. Debart, P. Enjalbert, M. Lescot, Multimodal logic programming using equational and order-sorted logic, Theoret. Comput. Sci. 105 (1992) 141–166.
[16] J.J. Elgot-Drapkin, Step-logic and the three-wise-men problem, in: AAAI, 1991, pp. 412–417.
[17] R. Fagin, J.Y. Halpern, Y. Moses, M.Y. Vardi, Reasoning About Knowledge, MIT Press, Cambridge, MA, 1995.
[18] L. Fariñas del Cerro, MOLOG: a system that extends PROLOG with modal logic, New Generation Comput. 4 (1986) 35–50.
[19] L. Fariñas del Cerro, A. Herzig, MOLOG—a tool for non-classical logic programming, ⟨http://www.irit.fr/ACTIVITES/EQ_ALG/Herzig/molog.html, 1986⟩.
[20] M. Fisher, R. Owens, An introduction to executable modal and temporal logics, in: M. Fisher, R. Owens (Eds.), Executable Modal and Temporal Logics, IJCAI'93 Workshop, Springer, Berlin, 1995, pp. 1–20.
[21] M. Fitting, R.L. Mendelsohn, First-Order Modal Logic, Kluwer Academic Publishers, Dordrecht, MA, 1999.
[22] J.W. Garson, Quantification in modal logic, in: D. Gabbay, F. Guenthner (Eds.), Handbook of Philosophical Logic: Volume II: Extensions of Classical Logic, Reidel, Dordrecht, 1984, pp. 249–307.
[23] K. Konolige, Belief and incompleteness, Technical Report 319, SRI International, 1984.
[24] R.A. Kowalski, Predicate logic as a programming language, Inform. Process. 74 (1974) 569–574.
[25] A. Leitsch, The Resolution Calculus, Springer, Berlin, 1997.
[26] J.W. Lloyd, Foundations of Logic Programming, second ed., Springer, Berlin, 1987.
[27] J. McCarthy, First order theories of individual concepts and propositions, Mach. Intell. 9 (1979) 120–147.
[28] J.-J.Ch. Meyer, W. van der Hoek, Epistemic Logic for Computer Science and Artificial Intelligence, Cambridge University Press, Cambridge, MA, 1995.
[29] J. Minker, Logic and databases: a 20-year retrospective, in: Internat. Workshop LID'96, Verlag Springer, Berlin, 1996, pp. 3–57.
[30] L.A. Nguyen, Constructing the least models for positive modal logic programs, Fund. Inform. 42 (1) (2000) 29–60.
[31] L.A. Nguyen, A fixpoint semantics and an SLD-resolution calculus for modal logic programs, Fund. Inform. 55 (1) (2003) 63–100.
[32] L.A. Nguyen, Multimodal logic programming and its applications to modal deductive databases, Manuscript (served as a technical report), Available at ⟨http://www.mimuw.edu.pl/∼nguyen/papers.html⟩, 2003.
[33] L.A. Nguyen, MProlog: an extension of Prolog for modal logic programming, in: B. Demoen, V. Lifschitz (Eds.), Proc. of ICLP 2004, Lecture Notes in Computer Science, Vol. 3132, Springer, Berlin, 2004, pp. 469–470.
[34] L.A. Nguyen, The modal logic programming system MProlog, in: J.J. Alferes, J.A. Leite (Eds.), Proc. of JELIA 2004, Lecture Notes in Computer Science, Vol. 3229, Springer, Berlin, 2004, pp. 266–278.
[35] L.A. Nguyen, An SLD-resolution calculus for basic serial multimodal logics, in: D.V. Hung, M. Wirsing (Eds.), Proc. of ICTAC 2005, Lecture Notes in Computer Science, Vol. 3722, Springer, Berlin, 2005, pp. 151–165.
[36] L.A. Nguyen, On modal deductive databases, in: J. Eder, H.-M. Haav, A. Kalja, J. Penjam (Eds.), Proc. of ADBIS 2005, Lecture Notes in Computer Science, Vol. 3631, Springer, Berlin, 2005, pp. 43–57.
[37] L.A. Nguyen, Reasoning about epistemic states of agents by modal logic programming, in: F. Toni, P. Torroni (Eds.), Proc. of CLIMA VI, Lecture Notes in Artificial Intelligence, Vol. 3900, Springer, Berlin, 2006, pp. 37–56.
[38] A. Nonnengart, How to use modalities and sorts in Prolog, in: C. MacNish, D. Pearce, L.M. Pereira (Eds.), Logics in Artificial Intelligence, European Workshop, JELIA '94 York, UK, September 5–8, 1994, Proceedings, Lecture Notes in Computer Science, Vol. 838, Springer, Berlin, 1994, pp. 365–378.
[39] H.J. Ohlbach, A resolution calculus for modal logics, in: Proc. of CADE-88, Lecture Notes in Computer Science, Vol. 310, Springer, Berlin, 1988, pp. 500–516.
[40] M.A. Orgun, W. Ma, An overview of temporal and modal logic programming, in: D.M. Gabbay, H.J. Ohlbach (Eds.), Proc. First Internat. Conf. on Temporal Logic, Lecture Notes in Artificial Intelligence, Vol. 827, Springer, Berlin, 1994, pp. 445–479.
[41] M.H. van Emden, R.A. Kowalski, The semantics of predicate logic as a programming language, J. ACM 23 (4) (1976) 733–742.